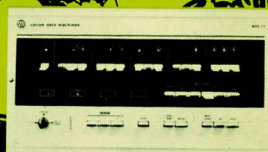
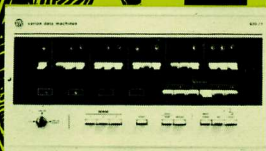


varian 620 100's computer handbook



**varian
data machines**

The Big Company in Small Computers

VARIAN 620-100 SERIES

COMPUTER HANDBOOK

Specifications Subject to Change Without Notice



varian data machines/a varian subsidiary
2722 michelson drive/irvine, california 92664

© 1972

98 A 9905 003

APRIL 1972

INTRODUCING...

THE VARIAN 620-100 SERIES COMPUTERS

The **Varian 620-100 Series Computers** are system-oriented, general-purpose minicomputers. They are designed for maximum performance in instrumentation, data acquisition, and communications systems, making them ideal for a variety of scientific, commercial, and industrial applications.

PART 1 of this Handbook describes the Varian 620/f-100 Computer. The Varian 620/L-100 Computer is discussed, insofar as its design differs from that of the 620/f-100, in **PART 2**.

The Varian 620-100 Series Computers feature:

Efficient Operation -- Operates on 16-bit words with a full-cycle execution time of 750 nanoseconds for the 620/f-100 and 950 nanoseconds for the 620/L-100.

Large Instruction Set -- The 620/f-100 recognizes 150 arithmetic, decision, and control instructions, many of which can be microcoded to extend the effective instruction repertoire into the hundreds. The 620/L-100 handles in excess of 133 instructions.

Modular Core Memory -- Can be expanded in 4,096-word (4K) increments, from the minimum of 4K to a maximum of 32K.

Multiple Addressing Modes -- Both the 620/f-100 and 620/L-100 include direct, multilevel indirect, immediate, indexed/

indirect, relative, and extended addressing. In addition, the 620/f-100 includes preindexing and postindexing.

Effective Input/Output -- Includes a full complement of peripheral controllers and devices and three types of I/O operations: program-controlled, direct memory access, and priority memory access.

Extensive Software -- Complete package includes the DAS assemblers, binary load/dump (BLD II) program, AID II for debugging, MAINTAIN II for troubleshooting FORTRAN IV, BASIC, the business-oriented RPG IV compiler, the Varian 620 Master Operating System (MOS), and a comprehensive mathematical and utility subroutine library. The 620/f-100 also includes VORTEX, the most powerful operating system available to today's minicomputer users. VORTEX offers a powerful real-time, multitasking operating system, providing complete rotating memory file management, concurrent foreground/background processing, program overlay, and simplified I/O operation.

Other recently released software includes Varian's PERT and BEST.

PERT performs scheduling analyses by the program evaluation and review technique. The PERT system combines the elements of a job schedule into a network and represents interdependencies and interrelationships in graphic form. Thus, PERT is

FOREWORD

a visual aid to project control managers. Using PERT, project managers can measure progress, determine task responsibility, and identify both slack periods and potential bottlenecks in complex projects spanning a year or more.

PERT allows the user to create, update, and reschedule PERT networks directly from a terminal, thus avoiding costly and time-consuming batch-oriented methods. The PERT technique has been adopted and used by a wide variety of governmental agencies and industries. It is effective whenever complex programs are to be completed within specified limits of time and money.

BEST (Basic Executive Scheduler and Timekeeper), a real-time monitor for Varian 620 series computers, allows a variable number of core-resident routines to operate concurrently within a relative-priority system.

The scheduling algorithm allows a program to be run at a specific time of day, after a specific interval, or at the next opportunity.

BEST also provides the following secondary functions: DEBUG, arithmetic package, and data manipulation package.

Scope of the Handbook

This Handbook explains the Varian 620-100 series computer systems and their capabilities. It contains both hardware and software information required for operating, interfacing, and programming the computer. The organization of the Handbook allows the experienced programmer or systems engineer to refer directly to the material that will aid him in understanding specific, task-related elements of the

system. The less experienced reader will find the Handbook a valuable introduction to computer concepts and technology.

Varian Data Machines invites suggestions on the contents of the Handbook so that it can be made as useful as possible to the reader. Direct any comments to the Publications Department, Varian Data Machines, 2722 Michelson Drive, Irvine, California 92664.

Support Documentation

To ensure full utilization of the Varian 620-100 series computers, complete documentation (including this Handbook) is provided with each system.

Maintenance and service information, including a detailed theory of operation, is contained in the Varian 620-100 Maintenance Manuals. Volume 2 of these manuals comprises assembly drawings, logic diagrams, and parts lists; they are assembled when the equipment is shipped to reflect current hardware revisions and the user's system configuration.

Other documentation provided with Varian 620-100 systems includes: technical manuals for internal computer options and peripheral devices; the Varian 620 Test Programs Manual, which describes operation of the MAINTAIN II system for diagnosing hardware malfunctions; programming manuals explaining software systems not discussed in detail in this Handbook, e.g., FORTRAN IV, BASIC, MOS, VORTEX, etc., and manuals and other documents relating to equipment designed to meet special, application requirements.

TABLE OF CONTENTS

PART 1 - VARIAN 620/f-100 COMPUTER

SECTION 1 - A COMPUTER IS . . .

Anatomy of a Computer.....	1-1
Numbers for Computers.....	1-3
Computer Arithmetic.....	1-4
Computer Logic.....	1-6
Programming the Computer.....	1-7
Operating the Computer.....	1-9
Computer Applications.....	1-10
Glossary	1-11
Bibliography.....	1-14

SECTION 2 - VARIAN 620/f-100 COMPUTER

SECTION 3 - SPECIFICATIONS

SECTION 4 - CENTRAL PROCESSING UNIT

CPU Section	4-1
Memory Interface.....	4-4
I/O Interface.....	4-4
Standard Features.....	4-4

SECTION 5 - MEMORY

Memory Design.....	5-1
Memory Operatio.....	5-1
Circuit Cards.....	5-1
Interfacing and Timing.....	5-2
Wrap-Around Addressing.....	5-2
Specifications.....	5-2

CONTENTS

SECTION 6 - COMPUTER OPTIONS

Teletype Controller 6-1

Automatic Bootstrap Loader 6-2

SECTION 7 - INPUT/OUTPUT SYSTEM

I/O Options 7-1

Organization 7-1

I/O Operation 7-6

Device Addresses 7-10

SECTION 8 - I/O OPTIONS

Priority Interrupt Module 8-1

Buffer Interlace Controller 8-4

Priority Memory Access 8-7

SECTION 9 - PERIPHERALS AND I/O INTERFACES

Teletypes 9-1

High-Speed Paper Tape Equipment..... 9-4

Magnetic Tape Equipment 9-6

Disc Memories 9-11

Drum Memories 9-14

Line Printer Equipment..... 9-17

Oscilloscope Display System..... 9-20

Punched Card Equipment 9-22

Digital Plotter..... 9-25

Buffered I/O Controller 9-27

Relay I/O Module 9-29

Digital I/O Controller 9-31

Analog/Digital Conversion Equipment..... 9-33

Data Communications Equipment..... 9-37

Universal Controller 9-48

Disc Memory System 9-50

SECTION 10 - PHYSICAL CONFIGURATION

System Planning 10-1

Physical Description..... 10-2

SECTION 11 - SYSTEM INTERCONNECTION

Memory Interconnection.....	11-1
I/O Expansion Interconnection.....	11-1
Power Supply Interconnection.....	11-1
Teletype Interconnection	11-7
System Interrupt Interconnection.....	11-7
PIM Interconnection.....	11-10
BIC Interconnection.....	11-10
PMA Interconnection.....	11-10
Cable Pin Assignments.....	11-10

SECTION 12 - SYSTEM OPERATION

Program Execution.....	12-1
Switches and Indicators.....	12-1
Manual Operations.....	12-6

SECTION 13 - MAINTENANCE

Routine Maintenance.....	13-1
Troubleshooting.....	13-2

SECTION 14 - COMPUTER WORD FORMATS

Data Word.....	14-1
Address Word.....	14-1
Instruction Word.....	14-2
Instruction Format Summary.....	14-6

SECTION 15 - ADDRESSING MODES

Direct Addressing.....	15-1
Indirect Addressing.....	15-1
Relative Addressing.....	15-1
Indexed Addressing.....	15-2
Extended Addressing.....	15-4
Immediate Addressing.....	15-4

CONTENTS

SECTION 16 - INSTRUCTION SET

Load/Store Instructions.....	16-2
Shift/Rotation Instructions.....	16-6
Register Transfer/Modification Instructions	16-10
Arithmetic Instructions.....	16-17
Logic Instructions.....	16-21
Jumper Instructions.....	16-24
Jump-and-Mark Instructions.....	16-32
Execution Instructions.....	16-38
Control Instructions.....	16-44
I/O Instructions.....	16-45

SECTION 17 - VARIAN 620 ASSEMBLERS

Character Set.....	17-1
Format.....	17-2
Computer Instructions.....	17-4
Assembler Directives.....	17-9
Symbol and Expression Modes.....	17-23
Relocatability Rules.....	17-23
Assembler Input Media.....	17-25
Assembler Output Listing.....	17-26
Error Messages.....	17-26
Operating the Assembler.....	17-28

SECTION 18 - BINARY LOAD/DUMP PROGRAM

Loading the Bootstrap Routine.....	18-1
Loading the BLD II Program.....	18-3
Loading an Object Program.....	18-5
Punching Program Tapes	18-7
Punching Memory Contents.....	18-7

SECTION 19 - AID II DEBUGGING PROGRAM

Loading AID II.....	19-1
Register and Memory Modification	19-1
Paper Tape Handling.....	19-3
Magnetic Tape Handling.....	19-4
Error Message and Correction.....	19-5

SECTION 20 - SOURCE PROGRAM EDITOR

Loading EDIT	20-1
EDIT Command.....	20-2
Usage Example.....	20-5
Error Message.....	20-5

SECTION 21 - MATHEMATICAL SUBROUTINES

Fixed-Point Arithmetic	21-1
Floating-Point Arithmetic.....	21-2
Arithmetic Functions.....	21-2
Conversion	21-4

SECTION 22 - MAINTAIN II TEST PROGRAMS

Test Executive Program.....	22-2
Test Programs.....	22-2

SECTION 23 - MASTER OPERATING SYSTEM

SECTION 24 - FORTRAN IV

Programming With FORTRAN IV.....	24-1
Operational Features	24-2

SECTION 25 - BASIC LANGUAGE

Programming With BASIC.....	25-1
Operational Features	25-2

SECTION 26 - RPG IV (REPORT PROGRAM GENERATOR)

Programming With RPG IV.....	26-1
Operational Features	26-2

SECTION 27 - VORTEX

CONTENTS

PART 2 - VARIAN 620/L-100 COMPUTER

SECTION 28 - VARIAN 620/L-100 COMPUTER

SECTION 29 - SPECIFICATIONS

SECTION 30 - CENTRAL PROCESSING UNIT

Control Section.....	30-1
Arithmetic/Logic Section	30-1
Operation Registers	30-3
Auxiliary Registers.....	30-3
Internal Buses.....	30-3
Information Transfer	30-4
Timing.....	30-6
Standard Features	30-13

SECTION 31 - MEMORY

Memory Design	31-1
Memory Operations.....	31-1
Circuit Cards	31-1
Interfacing and Timing	31-2
Wrap-Around Addressing.....	31-6
Bootstrap Loader Protection	31-6
Specifications.....	31-6

SECTION 32 - COMPUTER OPTIONS

Bootstrap Loader Protection	32-1
-----------------------------------	------

SECTION 33 - PHYSICAL CONFIGURATION

Circuit Cards	33-1
System Planning.....	33-1
Physical Description.....	33-2
Memory Expansion.....	33-7
I/O Expansion	33-7
Power Supply.....	33-13

SECTION 34 - SYSTEM INTERCONNECTION

Connector Panels.....	34-1
Memory Expansion Interconnection.....	34-4
I/O Expansion Interconnection.....	34-5
Teletype Interconnection.....	34-7
System Interrupt Interconnection.....	34-8
PIM Interconnection.....	34-9
BIC Interconnection.....	34-10

SECTION 35 - SYSTEM OPERATION

Program Execution.....	35-1
Switches and Indicators.....	35-1
Manual Operations	35-4

CONTENTS

APPENDIX A ALPHABETIC INDEX OF INSTRUCTIONS

APPENDIX B NUMBER SYSTEMS

APPENDIX C POWERS OF TWO

APPENDIX D OCTAL/DECIMAL INTEGER CONVERSIONS

APPENDIX E OCTAL/DECIMAL FRACTION CONVERSIONS

APPENDIX F STANDARD CHARACTER CODES

APPENDIX G TELETYPE CHARACTER CODES

LIST OF ILLUSTRATIONS

1-1	Typical Digital Computer System	1-2
1-2	Flowchart for Sample Program	1-9
2-1	Fully Expanded Varian 620/f-100 Computer Systems	2-2
4-1	Varian 620/f-100 Block Diagram	4-2
5-1	Varian 620/f-100 Memory Block Diagram	5-3
5-2	Memory Interface Waveforms	5-4
7-1	Typical 620/f-100 I/O System Block Diagram	7-2
7-2	Typical E Bus Line Configuration	7-3
7-3	Typical Control Line From the Computer	7-4
7-4	Typical Control Line to the Computer	7-5
7-5.	External Control Timing	7-9
7-6.	Sense Response Timing	7-10
7-7.	Data-Transfer-In Timing	7-11
7-8.	Data-Transfer-Out Timing	7-12
8-1	PMA/Controller Interface	8-10
8-2	PMA, DMA, Control Line	8-11
8-3	PMA Functional Block Diagram	8-12
8-4	Typical PMA Control Line Out	8-14
8-5	Typical PMA 16-Bit Bidirectional Bus Line	8-15
8-6	Typical PMA Control Line In or Address Line	8-16
8-7	BTC Block Diagram	8-17
10-1	Mainframe With Control Panel Extended	10-3
10-2	Top View of CPU Tray	10-5
10-3	Typical Circuit Card in CPU Tray	10-7
10-4	Power Supply Panels	10-7
10-5	Memory Chassis	10-8
10-6	Memory Chassis Connector Panel	10-8
10-7	Memory Chassis Cable Connectors	10-10
10-8	I/O Expansion Chassis	10-10
11-1	Chassis Connectors	11-2
11-2	Mainframe/Memory Chassis Interconnection	11-3
11-3	I/O Cable Installation in 32K Systems	11-4
11-4	I/O Cable Installation in 16K System	11-5
11-5	Power Interconnection Without Expansion Supply	11-6
11-6	Power Interconnection With Expansion Supply	11-8
11-7	Typical System Interrupt Priority Connections	11-9
12-1	Varian 620/f-100 Control Panel	12-2
14-1	Instruction Processing, Simplified Flow	14-3
14-2	Jump Instruction, General Flow	14-7
14-3	Jump-and Mark Instruction, General Flow	14-8
14-4	Execution Instruction, General Flow	14-9
15-1	Preindexing and Postindexing	15-3
17-1	Manipulation of Expression and Symbol Modes	17-24

CONTENTS

18-1	BLD II Tape Format (Bootstrap-Loadable)	18-3
18-2	Object Program Tape Format	18-6
22-1	MAINTAIN II Test Program System	22-1
23-1	MOS Partitions and Flow	23-2
30-1	Varian 620/L-100 Computer Organization	30-2
30-2	Basic Clock Waveforms	30-7
30-3	Example of Modified Clock Sequence	30-9
30-4	Sequence for Accessing an Operand in Memory	30-11
30-5	Sequence for Storing an Operand in Memory	30-12
30-6	Sequence for Indirect Operand Access	30-14
31-1	Memory Block Diagram	31-3
31-2	Memory Full-Cycle Timing	31-4
31-3	Memory Half-Cycle Timing	31-5
33-1	Varian 620/L-100 Mainframe	33-3
33-2	Control Panel Opened To Expose Backplane	33-4
33-3	Mainframe Circuit Card Locations	33-6
33-4	Varian 620/L-100 Expansion Chassis	33-8
33-5	All-Memory Expansion Card Locations	33-10
33-6	Memory-and-I/O Expansion Card Locations	33-11
33-7	All-I/O Expansion Card Locations	33-12
33-8	Power Supply Top Panel	33-13
33-9	Power Supply Bottom Panel	33-14
34-1	Mainframe Connector Panel	34-1
34-2	Expansion Chassis Connector Panel	34-2
34-3	Memory Expansion Interconnection	34-3
34-4	I/O Expansion Interconnection	34-6
34-5	Typical System Interrupt Priority Connection	34-9
35-1	Varian 620/L-100 Control Panel	35-2

LIST OF TABLES

4-1	MP Specifications	4-6
4-2	PF/R Specification	4-7
4-3	M/D Specifications	4-9
4-4	RTC Specifications	4-10
5-1	Memory Specifications	5-4
6-1	Teletype Controller Specifications	6-1
6-2	ABL Specifications	6-3
7-1	E Bus and I/O Control Signals	7-7
7-2	Standard Device Address	7-11
8-1	PIM Specifications	8-3
8-2	PIM Instructions	8-4
8-3	BIC Specifications	8-5
8-4	BIC Instructions	8-7
8-5	PMA Specifications	8-8
8-6	PMA Instructions	8-18
9-1	Teletype Controller Specifications	9-2
9-2	Teletype Controller Instructions	9-3
9-3	High-Speed Paper Tape Controller Specifications	9-4
9-4	High-Speed Paper Tape Controller Instructions	9-5
9-5	Seven-Track Magnetic Tape Controller Specifications	9-7
9-6	Seven-Track Magnetic Tape Controller Instructions	9-8
9-7	Nine-Track Magnetic Tape Controller Specifications	9-9
9-8	Nine-Track Magnetic Tape Controller Instructions	9-10
9-9	Model 620-38 Disc Memory Specifications	9-11
9-10	Model 620-38 Disc Memory Instructions	9-12
9-11	Disc Memory Controller Instructions	9-13
9-12	Drum Memory System Specifications	9-15
9-13	Drum Memory Controller Instructions	9-16
9-14	Line Printer Equipment Specifications	9-17
9-15	Line Printer Controller Instructions	9-19
9-16	Oscilloscope Display System Specifications	9-20
9-17	Oscilloscope Controller Instructions	9-21
9-18	Card Reader Controller Specifications	9-22
9-19	Card Reader Controller Instructions	9-23
9-20	Card Punch Controller Specifications	9-24
9-21	Card Punch Controller Instructions	9-24
9-22	Digital Plotter System Specifications	9-25
9-23	Digital Plotter Controller Instructions	9-26
9-24	Buffered I/O Controller Specifications	9-27
9-25	Buffered I/O Controller Instructions	9-28
9-26	Relay I/O Module Specifications	9-29
9-27	Relay I/O Module Instructions	9-30
9-28	Digital I/O Controller Specifications	9-31
9-29	Digital I/O Controller Instructions	9-32

CONTENTS

9-30	Analog Input Module Specifications.....	9-34
9-31	Analog Input Module Instructions.....	9-35
9-32	Analog Output Module Specifications.....	9-35
9-33	Analog Output Module Instructions.....	9-36
9-34	Model 620-65 Dataset Controller Specifications.....	9-38
9-35	Model 620-65 Dataset Controller Instructions.....	9-39
9-36	Model 620-66 Dataset Controller Specifications.....	9-39
9-37	Model 620-66 Dataset Controller Instructions.....	9-40
9-38	ACU Controller Specifications.....	9-41
9-39	ACU Controller Instructions.....	9-42
9-40	Data Communications Controller Specifications.....	9-43
9-41	Data Communications Controller Instructions.....	9-44
9-42	Varian 620/DC Specifications.....	9-45
9-43	Varian 620/DC Instructions.....	9-46
9-44	Universal Controller Specifications.....	9-48
9-45	Model 620-35, -35A Disc Memory Specifications.....	9-50
9-46	Disc Memory Controller Instructions.....	9-51
9-47	Model 620-36, -36A Disc Memory Specifications.....	9-52
9-48	Disc Memory Controller Specifications.....	9-53
10-1	Card Locations in CPU Tray.....	10-6
10-2	Memory Card Assignments.....	10-9
11-1	Memory Address Cable.....	11-11
11-2	Memory Data Cable.....	11-12
11-3	I/O Cable.....	11-13
11-4	I/O Expansion Cable.....	11-14
11-5	CPU Power Cable.....	11-15
11-6	Memory Power Cable.....	11-15
11-7	I/O Power Cable.....	11-15
11-8	Teletype Cable.....	11-15
12-1	Bootstrap Loader Routines.....	12-7
13-1	Recommended Test Equipment.....	13-2
15-1	Instruction Addressing-Mode Specifications.....	15-2
16-1	Instruction Groups.....	16-1
17-1	Assembler Instruction Type Characteristics.....	17-6
17-2	Summary of Assembler Instruction Types.....	17-6
17-3	Directives Recognized by DAS Assemblers.....	17-10
17-4	DAS Symbol Table Capacities.....	17-11
17-5	Standard DAS 8A Location Counters.....	17-13
17-6	DAS Error Codes.....	17-27
17-7	Acceptable I/O Devices.....	17-29
18-1	Bootstrap Loader Routines.....	18-2
18-2	BLD II SENSE Switch Options.....	18-4
19-1	AID II Register/Memory Modification Commands.....	19-2
19-2	AID II Paper Tape Commands.....	19-4
19-3	AID II Magnetic Tape Commands.....	19-5
20-1	EDIT Commands.....	20-2
20-2	Teletype Key EDIT Functions.....	20-4
30-1	Varian 620/L-100 System Clocks.....	30-8

30-2	M/D Specifications	30-15
30-3	MP Specifications	30-17
30-4	MP Instructions	30-18
30-5	RTC Specifications	30-19
30-6	RTC Instructions	30-20
30-7	PF/R Specifications	30-21
32-1	Memory Specifications	31-6
32-1	Bootstrap Protection Specifications	32-1
33-1	Varian 620/L-100 Circuit Cards	33-1
33-2	Mainframe Card Slot Assignments	33-5
33-3	All-Memory Expansion Card Slot Assignments	33-9
34-1	Power Supply Terminal Board Pin Assignments	34-2
34-2	J30 DC Power Cable Pin Assignments	34-3
34-3	Memory Expansion Cable Pin Assignments	34-4
34-4	I/O Expansion Cable Pin Assignments	34-6
34-5	33 ASR Teletype Cable Pin Assignments	34-8
34-6	35 ASR and 35 KSR Teletype Cable Pin Assignments	34-8
34-7	Interrupt Line Pin Assignments	34-9
35-1	Bootstrap Loader Routines	35-4

SECTION 1 - A COMPUTER IS . . .

A computer is . . .

- *A reporter*
- *An accountant*
- *A mathematician*
- *An efficiency expert*
- *A solver of problems*

And a computer has . . .

- *An unerring memory*
- *Lightening speed*
- *Infinite patience*
- *An even temperament*
- *An appetite for boring details*

But, for all its virtues, the computer is merely a machine. It can solve a problem with great speed and accuracy, but only if someone, a human being, has analyzed and prepared the problem and told the machine **exactly** what must be done, how to do it, and when to do it, in a language the machine can understand.

The way that a computer works can be compared to the thoughts and actions of a man when he adds a column of figures. He needs sheets of paper on which to write the problem and the answer to the problem. He needs his eyes to read the problem, and his hand to hold the pencil. He needs to have stored in his memory a method for adding numbers; he must know how to read and write. And, he needs to have the power to bring all of these elements into play.

So it is with the computer.

A computer's "sheets of paper" are called peripheral devices; its "eyes and hands", an input/output (I/O) system. The computer's memory serves the same purpose as the man's memory, and its central processing unit (CPU) is like the portion of the man's brain that controls his actions, his central nervous system.

The "program" for reading the numbers, performing the arithmetic operations, and writing the results is stored in the man's memory when he is taught to read, write, and add. The computer is programmed in the same way by a set of instructions that, when executed in sequence, solves a complex problem by a series of simple arithmetic operations.

Man has an important role in solving the problem. . . he must program the computer.

Anatomy of a Computer

There are two types of computer: *analog* and *digital*.

The analog computer is a specialized computing device, electronically wired (programmed) to perform a specific task. Its inputs are, generally, quantities to be measured rather than counted. The computer acts on the inputs, performing a number of mathematical operations or solving an equation, and plots the results on a graph.

A COMPUTER IS . . .

The **digital computer**, on the other hand, is general-purpose in nature; it counts rather than measures quantities. The digital computer is, in many respects, a high-speed automatic calculator, and it not only follows instructions, but can remember whole series of them. Thus, this type of computer can be programmed to perform a variety of computational and control tasks.

A digital computer can also serve as a special-purpose (dedicated) computer, repetitively executing only one operation (e.g., controlling a chemical process or preparing inventory records). Its general-purpose capabilities remain and can be exploited simply by introducing other programs (software).

The **Varian 620-100 series computers** are digital computers. Hence, this section of the Handbook deals entirely with digital computing techniques.

A digital computer system can be divided into four basic sections as shown in figure 1-1.

- The **Central Processing Unit** controls and coordinates computer operations for solving the problem.
- The **Memory** stores the program, the problem, and the solution.
- The **Input Section** provides coded information and instructions to the computer.
- The **Output Section** gives the user the processed data (the solution to the problem).

Central Processing Unit

The central processing unit (CPU) has three distinct jobs: control computer operations, make decisions (i.e., add or subtract), and tell the operator what it is doing.

The **control section** makes all computer decisions and controls the manipulation of the data fed to it. It does this by interpreting and executing the instructions and

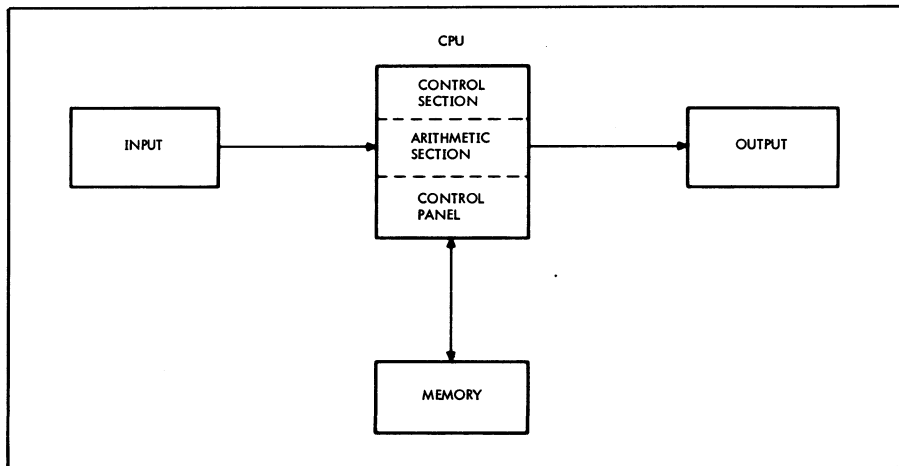


Figure 1-1. Typical Digital Computer System

information read from memory or received from the input unit.

The **arithmetic section** does the actual work in solving the problem using basic arithmetic operations. It also manipulates data under the supervision of the control section. The arithmetic section usually contains registers (accumulators) that hold the data and the results of its calculations, and logic (electronic circuitry) that allows the data in the registers to be combined with information transferred from memory or input devices.

The **control panel** gives the operator direct access to and control over CPU operations and memory. Switches and indicators permit him to examine or change the contents of memory or determine what is happening in the CPU and how the program is progressing. The control panel and a keyboard input device (e.g., a Teletype) are often grouped together under the term *console*.

Memory

The memory stores the instructions that direct the actions of the computer and the data to be acted upon, either numerical values (operands) for use in solving the problem, or constants that remain in memory and are referred to by the computer when necessary to the solution of the problem.

Computer memory has a number of individual locations (addresses), each capable of storing a computer word of information. Each address is uniquely designated so that specific information can be conveniently loaded and retrieved. Reading out the information stored in memory does not destroy the data; the same information can be referred to as long as changes are not specifically requested by the program or the operator.

The basic computer memory is, usually, a group of magnetic cores, collectively called **primary storage**. Magnetic tapes are a type of **secondary storage**; data read from or to a tape can supplement the capacity of basic memory. Magnetic drums (and discs), another type of secondary storage, consist of rotating metal drums (discs) around which are several tracks for recording data.

Input Section

The input section of a digital computer receives instructions and data from input devices, e.g., punched card readers, Teletype keyboards, magnetic tape units, disc or drum devices, etc. This section translates the information received from these devices into a form that memory can accept and store.

Output Section

The output section translates finished, processed data (answers) from the CPU into a form that can be accepted by output devices, e.g., card punches, line printers, magnetic tape units, etc., and transmits the translated data to these devices.

Note that some "peripheral" devices, e.g., magnetic tape units or disc devices, can function both as input and as output devices.

Numbers for Computers

Digital computers deal only with numerical values. Data to be manipulated, therefore, must be translated into some "digital" form. The **binary (radix, or base, of two) number system** is used because digital computers, whether large or small, are collections of electronic components, which have basic "on-off" characteristics. A relay is either opened or closed; magnetic materials (tape or cores) are magnetized

A COMPUTER IS . . .

in one direction or another; a vacuum tube or transistor is either fully conducting or not conducting; an electrical pulse may be transmitted at a given time or not transmitted.

In the computer, decimal input data are converted to binary numbers, computations are made using binary arithmetic operations, and output data are converted from binary back to decimal. The binary system not only can represent decimal numbers but, by coding, it can be made to represent letters of the alphabet as well.

Refer to appendix B for detailed information about the binary, and other, number systems. The following information is given to acquaint the reader with these systems as they apply to the Varian 620-100 computers.

Words in the Varian 620-100 computers comprise 16 bits expressed as octal (radix of eight) values. The octal system assigns numerical values to pure binary forms, which are divided into groups of three bits, each group representing a single octal digit. There are eight possible configurations:

Binary	Octal
000 =	0
001 =	1
010 =	2
011 =	3
100 =	4
101 =	5
110 =	6
111 =	7

The conversion from the binary configuration to the octal equivalent (appendix B) in the Varian 620-100 computers is illustrated in the example below.

Number conventions used throughout this handbook are: (1) binary numbers are divided into octal groupings, (2) octal numbers are prefixed with zero, and (3) decimal numbers have no prefix.

Computer Arithmetic

One of the advantages of the binary number system is the ease with which arithmetic problems can be solved. Few rules, and no tables, are required to solve any addition, subtraction, multiplication, or division problem (appendix B). Complex problems are first broken down into a series of simple binary arithmetic operations.

Binary Addition

The rules of binary addition are:

- a. $0 + 0 = 0$
- b. $0 + 1 = 1$
- c. $1 + 0 = 1$
- d. $1 + 1 = 0$ with 1 carried to the left one position

Bit Position

15*	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0**
0	0	1	0	1	0	0	0	0	1	1	0	1	0	1	1
0		2		4				1		5				3	

* Bit 15 = most significant bit (MSB). ** Bit 0 = least significant bit (LSB).

Binary Subtraction

Four rules apply to binary subtraction:

- a. $0 - 0 = 0$
- b. $1 - 1 = 0$
- c. $1 - 0 = 1$
- d. $0 - 1 = 1$ with 1 borrowed from the left

Subtraction is also possible using **complements**. Since there are only two digits in the binary system, there can be only two complements. To find the one's complement of binary numbers, **change all ones to zeros and all zeros to ones** (e.g., the complement of 1011 is 0100). To subtract, simply add the one's complement of the minuend to the binary subtrahend.

Binary Multiplication

As in the decimal system, binary multiplication is a series of shifts and additions. The following rules apply:

- a. $0 \times 0 = 0$
- b. $0 \times 1 = 0$
- c. $1 \times 0 = 0$
- d. $1 \times 1 = 1$

No carries are considered. Each digit of the multiplier is examined, and, if a one is found, the multiplicand is added to the result. When a zero is found in the multiplier, zeros are added to the result. The multiplicand is shifted left one digit for each multiplier digit.

Binary Division

Binary numbers can be divided by the application of the concepts of binary addition, subtraction, and multiplication. The dividend is inspected for the first group of digits from which the divisor can be subtracted once. A one is placed in the quotient over the last digit of the dividend group. This is continued with zeros appearing in the quotient where a subtraction is not possible after the next dividend digit is brought down to form the least significant digit of the new dividend.

Two's Complement Arithmetic

The Varian 620-100 computers use the two's complement technique for arithmetic operations. To find the two's complement of a binary number, **replace all ones with zeros and all zeros with ones (one's complement)**, then add one to the least significant digit.

Negative numbers are represented in two's complement form, and subtraction is accomplished by complement addition.

The Varian 620-100 computers operate on single-precision (one-word) numbers that contain 15 bits plus a sign bit. The sign bit occupies the most significant bit position (bit 15); zero in this bit denotes a positive number, and one, a negative number.

The range of numbers processed by Varian 620-100 computers is:

Binary	Octal	Decimal	
0 111 111 111 111 111	077777	+ 32,767	+ Full Scale
0 000 000 000 000 000	000000	0	
1 111 111 111 111 111	177777	- 1	
1 000 000 000 000 000	100000	- 32,768	- Full Scale

The *precision* of the Varian 620-100 computers is thus to 16 binary digits. Precision is defined as a degree of accuracy; e.g., a four-position table, correctly computer, is accurate, but a six-position table containing an error is more precise although not accurate. Any degree of accuracy can be obtained by increasing the number of digits. For instance, the value of π is 3.14159 (to five decimal places, or six digits). A computer that could manipulate only four digits would carry 3.142; increasing the precision to five digits would result in greater accuracy (3.1416); and an increase to six digits results in even greater accuracy.

As shown above, the largest positive decimal number that can be processed by the Varian 620-100 computers using their 16-bit configuration (single-precision operation) is +32,767. Double-precision arithmetic techniques, in which two computer words (32 bits) represent one number, allow the processing of larger numbers.

Computer Logic

As noted earlier, the electronic components in digital computer systems are inherently binary. They are either on or off, or open or closed. The arithmetic unit of the computer is made up of many of these two-state devices called stages. The three basic types of stages are:

- a. The *gate*, which is a circuit switch that can be open or closed. It has several possible inputs, and generates an output signal if certain conditions are met by the input signal. There are two types of gates: (1) the AND gate, which generates an output only when all input signals are present, and (2) the OR gate, or buffer, which generates an output if any input signals are present.

- b. The bistable circuit, or flip-flop, which has two stable states: (1) set or one, and (2) reset or zero. Flip-flops can be constructed from vacuum tubes, transistors, or other active components. In effect, the set and reset states are similar to the open and closed states of an ordinary switch. Because of its stable nature in both states, a flip-flop can be used to store binary information, e.g., the sign of a number (plus or minus).
- c. The shift register, which can be used for temporary storage of digital data, converts data from serial to parallel or vice versa, and generates pulse patterns. A shift register comprises a number of flip-flops connected so that each flip-flop transfers its information to the next when the input is pulsed.

Electronic circuits such as those described above are composed of active components (i.e., transistors) and passive components (i.e., capacitors and resistors), the appropriate interconnection of which provides a given electronic (logic) function. Miniaturization of these circuit elements has resulted in the integrated circuit (IC), which contains several circuit functions (inductance, capacitance, resistance, and amplification) in a single element.

Magnetic materials are used in digital computers as both memory and logical devices. These materials include a group known as ferrites, which are ceramic-like magnetic substances derived from metal oxides. The electrons of the ferrite material of a magnetic core are affected by external magnetic fields. The direction of current flow through the core produces one of two magnetic field states, corresponding to zero and one. The core state remains

stable after the current source that created it is gone; hence, data can be safely stored until needed for computation.

IC design techniques and improved core memory packaging make possible the economy and compact size of the Varian 620-100 computers, while providing the capability of performing complex computational and control tasks.

Programming the Computer

Any computer accepts and executes a certain number of instructions (sometimes called commands). Such expressions can be plain English words or phrases, e.g., CLEAR AND ADD, MULTILPLY, STORE WORD, etc. However, because of the length of such expressions and the frequency of their use in computer programs, abbreviated mnemonic forms are often substituted. For example, the instruction SET OVERFLOW INDICATOR can have the mnemonic form SOF.

Since the computer responds only to numerical input, each instruction has a corresponding numerical code. This code exists in the computer as a combination of the binary digits 0 and 1. For simple machines, such codes can be expressed in full binary form. However, for most computers the codes are condensed to octal (base 8) or hexadecimal (base 16) numbers.

The list of instructions or codes that a given computer accepts and executes is called the instruction set (repertoire) of that computer. The binary codes for these instructions comprise the machine language for that computer, the only language the machine understands.

Four essential steps are required to effectively and efficiently program a given

computer: analysis, flowcharting, coding, and checkout and use.

Analysis of the Problem

In preparing a problem for the computer, the programmer must first clearly define the problem and a mathematical method for solving it. Usually a specific problem can be solved a number of different ways. The programmer must take into account the capabilities of the computer and have a thorough knowledge of the machine's language.

Flowcharting

As an aid to programming, the flowchart is invaluable. A flowchart consists of a series of connected geometric figures, each denoting a step in solving the problem. Each geometric shape has a particular logical significance, e.g., rectangles indicate computer processing steps, diamonds indicate decisions, etc. The geometric forms are annotated with explanations and connected with lines and arrows showing the sequence and direction of process flow.

Since the flowchart is a tool for analysis and clarification, it generally shows only enough information to ensure:

- a. Correct compilation of the list of instructions in the program
- b. Proper planning for and allocation of memory space

The completed flowchart should be examined to see that a program based on it will actually allow the computer to solve the problem. Alternative methods that might require less computer running time and/or memory space should be considered. When this has been done, the programmer

A COMPUTER IS . . .

is ready to start writing (coding) the program based on the flowchart.

Coding

Different computers not only use different languages, but have vocabularies of different lengths. Thus, it is impossible to write a program for a machine without complete understanding of the instruction set. The programmer must also know the word formats applicable to the particular machine, and he should have some knowledge of the machine itself.

With this knowledge and a comprehensive flowchart, the programmer writes the source program in symbolic notations suitable for translation by the assembler or compiler that is compatible with the computer.

Checkout and Use

Errors in the program can cause incorrect results or computer stoppage. The process of locating and correcting errors or omissions (bugs) in a program is called debugging.

Generally, the first step in debugging a program is to ensure that the program stored in the computer is the same as the program that was coded. To compare them, prepare and load the program a second time and compare the results of the second run with those of the first. (Of course, if the program was loaded from cards, tapes, or similar input media, the

same media should not be used for the debugging run. New input should be prepared.)

As a second step in the debugging process, the Varian 620-100 computers allow visual display of loaded data and instructions, and execution of the program step by step. Using these methods, the programmer can ensure that the program is properly coded and loaded and that the solution of the problem is within the capacity of the computer.

When these factors have all been carefully considered and the results (if any), of a program are still incorrect, a computer malfunction can be suspected. Programs designed to aid in the diagnosis of such problems can be loaded into the machine to determine the cause of the malfunction. Such programs, which perform on the machine the same function that debugging does on the program, are called diagnostics.

After the program is debugged, it can be used on the computer when required.

Sample Program

Problem: Sum the even integers from 2 to 100 on the Varian 620-100 computer.

The solution is defined by the operations and logical decisions illustrated in the flowchart (figure 1-2). The program, which is self-explanatory (refer to the Comments column), is coded as follows.

Address	Instruction	Symbolic Coding	Comments
004000	014014	LOOP LDA NEXT	Compute the first
004001	124012	ADD TWO	(next higher) even
004002	054012	STA NEXT	integer, add it to
004003	124012	ADD TOTL	the cumulative
004004	054011	STA TOTL	total, decrement
004005	014011	LDA CNTR	the loop counter
004006	124004	ADD ONE	by one, and, if
004007	054007	STA CNTR	the sum is not com-
004010	001004	JAN LOOP	plete, repeat.
004011	004000 R		
004012	000000	HLT	When done, halt.
004013	000001	ONE DATA 1	Amount to decre-
			ment the counter.
004014	000002	TWO DATA 2	Constant for com-
			puting next integer.
004015	000000	NEXT DATA 0	Next even integer.
004016	000000	TOTL DATA 0	Cumulative total.
004017	177716	CNTR DATA - 50	Negative constant
			for iterations.
	000000	END	

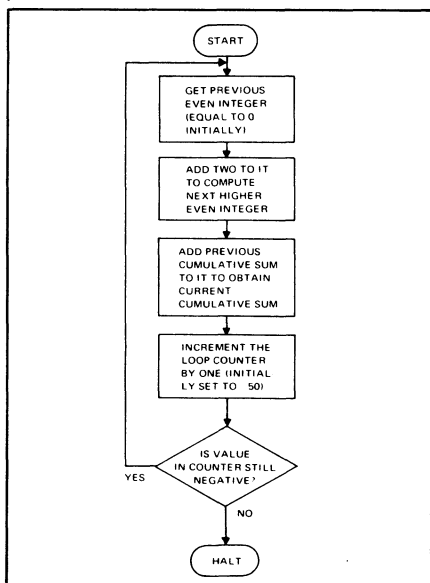


Figure 1-2. Flowchart for Sample Program

Operating the Computer

The computer operator enters the completed, debugged object program into computer memory through a peripheral input device, or through data entry switches on the control panel. The instructions are stored in sequential memory addresses, one instruction word to an address. When the computer operator presses the run switch on the control panel, the CPU "fetches" the first instruction and decodes the binary digits of the word to determine the action to be taken. The CPU then "executes" the instruction and returns to memory to fetch the next instruction. This fetch-and-execute process continues automatically until all the program instructions have been executed.

A COMPUTER IS . . .

The last instruction in a program is usually an instruction to halt computer operation, but it can also be an instruction to return to the start of a program. In this case, the CPU continuously "loops" through the program until the operator halts computer operations manually.

The Varian 620-100 computers recognize the instructions described in section 16 and indexed in appendix A. Each instruction has a name (mnemonic), which is translated to binary "machine" code by the assembly program (DAS Assemblers, section 17). The list of instruction mnemonics (symbolic listing) prepared by the programmer is designated the **source program**; the machine-code output produced by the assembler is referred to as the **object program**.

The object program is loaded in memory using a binary loader program (BLD II, section 18). A program that does not execute properly or that produces incorrect results can be analyzed with a "debugging" program (AID II, section 19), and existing programs can be changed with an "editing" program (EDIT, section 20). Equipment (hardware) malfunctions can be diagnosed with the aid of test programs (MAINTAIN II, section 22).

Detailed operating procedures for running programs on the Varian 620-100 computers, including a listing of the "bootstrap" loader program required for entering the object programs into the computer system, are given in section 10.

Computer Applications

Minicomputers, such as the Varian 620-100 computers, can be adapted to a wide range of applications, including business data pro-

cessing, industrial control, data communications, scientific research, and education.

The simplest computer system, the computer and a Teletype for input/output, is suitable for the research laboratory for simple computations. This basic system can be enhanced by the addition of peripherals that present results in graphic form, i.e., digital plotters and oscilloscope display systems. High-speed paper tape equipment (reader and punch) added to the basic system increases input/output efficiency and makes it possible to record computation results in a form suitable for later analysis.

Business and accounting applications require a large data base that can be efficiently stored, easily accessed, and promptly displayed or printed out. Adding disc memory and/or magnetic tape units (for bulk data storage) and a line printer (for output) to the basic computer-Teletype system meets these requirements. This system can be made even more flexible and efficient with the use of punched-card and/or high-speed paper-tape equipment.

Industrial process control is another important computer use. This application presents situations that require frequent sampling and interpretation of data as materials are being processed and as goods are being manufactured. Here again the basic computer-Teletype system comprises the heart of the system. Analog, digital, and relay I/O devices can be connected on the I/O bus for the input of signals from the processing/manufacturing area and for the output of signals to control devices or displays. Secondary data storage devices make it possible to record data for allied operations, i.e., accounting, production control, etc. If additional operator input is required, several Teletypes can be interfaced on the I/O bus.

A Varian 620-100 computer system can be configured to integrate and unify a communications network by adding the Varian 620/DC Data Communications System to the basic system. Used as a data concentrator, this system links and switches variable-speed modems and communications lines, adapting to a variety of communications devices and transfer rates. With the addition of disc or magnetic tape units, this system can be used as a preprocessor, organizing data for efficient, time-shared processing in larger computer systems.

Glossary

A

accumulator -- A part of the arithmetic unit of a digital computer where numbers are totaled (i.e., accumulated) and temporarily stored.

accuracy -- Freedom from error (*not* the same as precision).

adder -- A device for forming the sum of two numbers.

address -- 1. A label, name, or number identifying a register, memory location, or unit where information is stored. 2. The operand part of an instruction.

alphanumeric -- Coding system using letters and numbers.

arithmetic unit -- A device (or part of the computer) for performing basic operations including addition, subtraction, multiplication, and division.

B

base -- The radix of a numeric notation; the decimal system is to the base 10.

binary -- A numbering system to the base 2.

binary code -- A code of binary numbers (0 and 1).

binary digit -- Either 0 or 1.

bit -- A binary digit.

block -- A group of computer words.

branch -- A point in a program where there is a choice of steps; a program jump.

buffer -- Temporary storage.

bug -- Program (or computer) error or omission.

C

card -- A stiff paper, about 7-3/8 by 3-1/4 inches, which is punched or marked with data to be sensed electronically or visually.

card punch -- A machine for punching data on cards.

card reader -- A machine for sensing data contained in punched cards.

carry -- In addition, the overflow from one column to the next higher column.

character -- An elemental mark such as a letter, number, or special symbol.

code -- A group of characters representing data or instructions.

coding -- The translation of a program into actual machine instructions.

collate -- To arrange material (usually from more than one source) in a specific sequence.

A COMPUTER IS . . .

column -- A vertical array of characters.

compile -- To assemble subroutines into a program.

complement -- A quantity which, when added to another quantity, provides the least quantity of one more place (6 and 4 are complements in decimal notation).

conditional jump -- A point in a program where the computer skips a series of instructions if specified conditions are met.

control -- A device or signal that affects other devices or signals.

convert -- To change from one state to another.

counter -- A device for recording or registering a sequence of events.

cycle -- A complete series of steps or events.

D

decimal -- Pertaining to the base 10 numbering system.

diagnostic routine -- A program on the computer used for troubleshooting.

digit -- A single character or symbol.

double-precision number -- A number that is twice as long as a number normally handled.

dump -- To write the contents of memory in a form suitable for future use.

E

error -- Incorrect procedure, number, or result.

F

field -- A group of bits (less than one computer word) considered as a unit of information.

fixed point -- A system of notation in which the decimal or binary point is fixed with respect to one end of the numbers used.

flip-flop -- A bistable (two-state) device in which the two possible outputs can be labeled on-off, 0-1, left-right, etc.

floating point -- A system of notation which takes into account the varying location of the decimal or binary point by expressing each number as a sign, coefficient, and base power.

flowchart -- A program or routine expressed block diagram form.

G

gate -- A device that produces an output signal when certain specific conditions are met.

gray code -- A special binary code where successive numbers change by only one digit.

H

hold -- To retain information after copying.

I

indexing -- A method of address modification.

information -- Any form of data, such as an operand or instruction.

instruction -- A form of information that tells the computer what operation to

perform, where to get the operand, and what to do with the result.

I/O -- Input/output.

J

jump -- A change in a program in which the next sequential instruction is not executed; instead the computer skips one or more instructions.

L

language -- A complete, organized set of characters together with the rules for their use.

library -- A collection of subroutines, complete programs, etc.

logic -- A formal set of operational rules for computer device or circuit behavior.

loop -- A closed set or ring of instructions.

M

magnetic core -- A small torroid, with two stable states, used in high-speed memories.

magnetic drum -- A rotating cylinder, the magnetized surface of which stores information.

matrix -- An array of components formed by the intersection of vertical and horizontal elements.

memory -- A device or place for information storage.

mnemonic -- Relating to a system for remembering codes.

N

normalize -- To standardize, as to adjust numbers to floating-point format.

O

octal digit -- A digit in the octal (base 8) number system.

on-line operation -- Computer control of input or output data where the data are processed as soon as available.

operand -- A quantity used in or resulting from an operation.

operator -- One who runs programs on a computer.

P

parity check -- A method for checking the bits in a computer word to assure that none were lost.

peripheral -- Devices used with, but not an integral part of the computer, i.e., line printers, card readers/punches, etc.

precision -- The exactness of a measurement as opposed to the degree of accuracy or correctness.

program -- An ordered series of steps followed by the computer to solve a specific problem or class of problems.

programmer -- One who develops or writes a program.

R

radix point -- The point in a system of numbers separating integers from fractions.

A COMPUTER IS . . .

read -- To acquire information from some form of storage.

register -- A device for the temporary storage of information, such as a computer word.

round off -- To remove some of the less significant digits from a number, resulting in a smaller, less accurate number.

routine -- A set of computer instructions in a specific sequence.

S

shift -- To displace a number to the right or left a given number of positions.

sort -- To separate information into two or more classes or groups.

V

verifier -- A device for checking against errors.

W

word -- A group of alphanumeric characters with specific meaning, e.g., a computer word.

Bibliography

Lytel, Allen, **abc's of Computers**, Howard W. Sams & Co., Inc., Indianapolis, Ind., 1961.

Lytel, Allen, **abc's of Computer Programming**, Howard W. Sams & Co., Inc., Indianapolis, Ind., 1962.

Murphy, John S., **Basics of Digital Computer Programming**, John F. Rider Publisher, Inc., New York, 1964.

Sippl, Charles J., **Computer Dictionary and Handbook**, Howard W. Sams & Co., Inc., Indianapolis, Ind., 1966.

SECTION 2 - VARIAN 620/f-100 COMPUTER

The **Varian 620/f-100 Computer** (figure 2-1) is a general-purpose digital computer, designed for a variety of system applications.

The computer processes 16-bit words in a full-cycle execution time of 750 nanoseconds, or over 1.3 million cycles per second.

The instruction set of the Varian 620/f-100 comprises 150 instructions, many of which can be microcoded to extend the effective repertoire to several hundred instructions. Instructions include hardware multiplication and division, and indirect, immediate, and extended addressing operations.

Core memory can be expanded in 4,096-word (4K) increments, from a minimum of 4K to a maximum of 32K. Improved design allows the packaging of a fully expanded 32K system in two 10-1/2-inch-high, standard rack enclosures.

The central processing unit (CPU) features four user-accessible operation registers, four auxiliary registers, an overflow indicator, and convenient operator's control panel.

Six addressing modes can be implemented: direct, multilevel indirect, immediate, preindexing and postindexing, relative, and extended forms that permit direct addressing of any area of the fully expanded 32K system.

One power supply (contained in the mainframe) can furnish all the power required to maintain the maximum 32K system plus a number of peripheral controllers.

The CPU, memory protection, power failure/restart, real-time clock, hardware mul-

tiply/divide, power supply, and mainframe computer options are housed in the mainframe. The memory chassis accommodates either 16K of memory and up to 12 I/O controller cards, or 20K through 32K of memory. Systems with 20K to 32K of memory include an I/O expansion chassis with 12 I/O card slots. An additional 12 I/O card slots can be added by adding another half backplane. Additional I/O expansion is available in all configurations by adding I/O expansion chassis, I/O wiring backplanes, and expansion power supplies if necessary.

Mainframe options include: automatic bootstrap loader (ABL), Teletype controller (TC), and priority memory access (PMA).

System I/O options include: priority interrupt module (PIM), buffer interlace controller (BIC), and PMA. The PIM establishes eight levels of interrupt requests on the I/O bus in order of priority. The BIC implements the direct memory access (DMA) capabilities of the basic computer, permitting cycle-stealing I/O data transfers between memory and peripheral controllers at rates of up to 275,000 words per second. The PMA allows up to four external devices direct access to the 620/f-100 memory.

Varian offers a complete complement of peripheral devices and their controllers to simplify total system planning and installation. Available peripherals include: high-speed paper-tape equipment, magnetic-tape units, disc and drum memories, punched-card equipment, and a variety of analog, digital, and communications equipment.

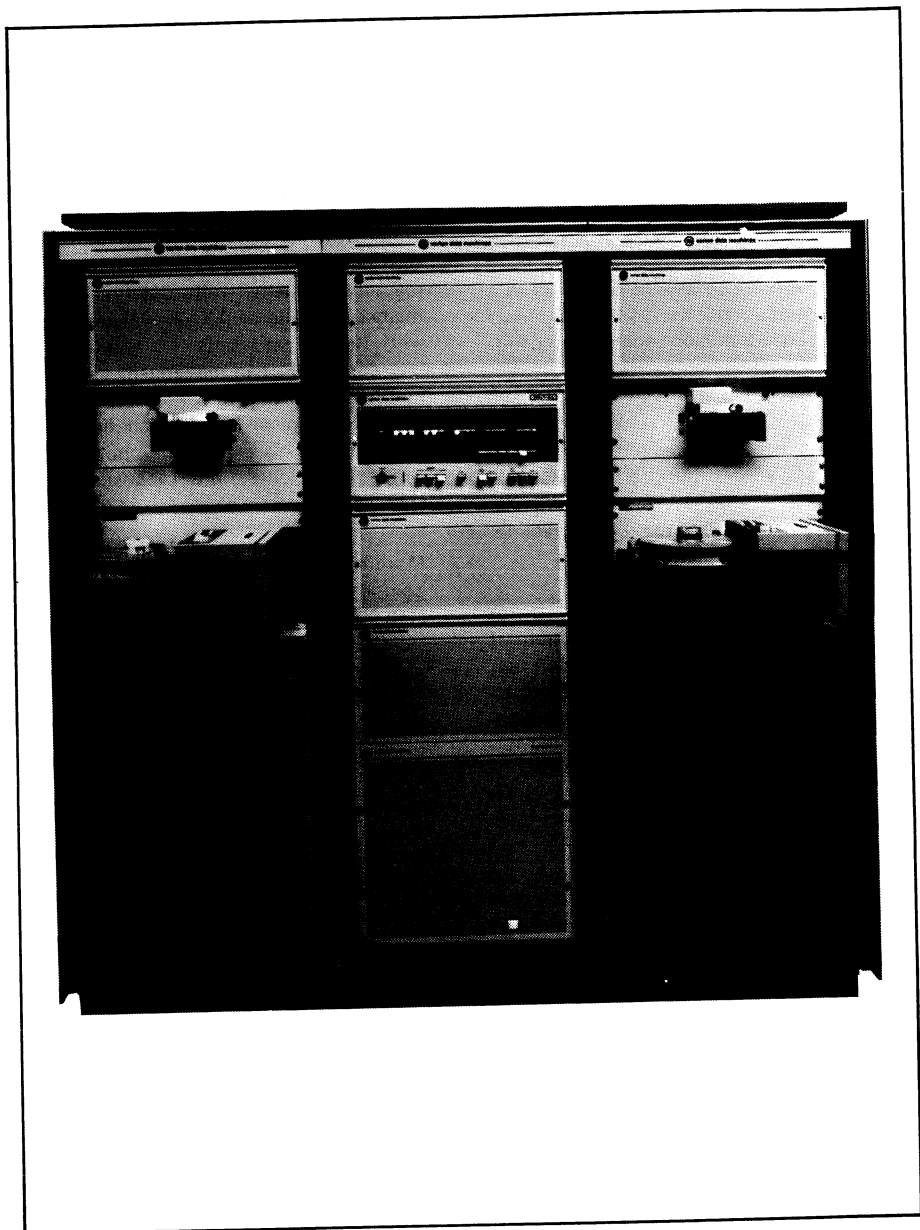


Figure 2-1. The Varian 620/f-100 Computer System

Standard software for the Varian 620/f-100 includes: the DAS symbolic assemblers, FORTRAN IV, BASIC, Master Operating System (MOS), the business-oriented RPG IV, AID II for debugging, MAINTAIN II for hardware troubleshooting, plus a complete library of mathematical and utility subroutines. Also available is the Varian Omnitask Real-Time EXecutive (VORTEX), which is a modular operating system for controlling, scheduling, and monitoring tasks in a real-time multiprogramming environment.

Varian's field-service organization provides a comprehensive program to aid the user of the Varian 620/f-100 in planning, installing, and maintaining the total system application. Service contracts cover necessary scheduled preventive maintenance and emergency repairs. VOICE, Varian's user organization, acts as a clearinghouse for user inquiries and suggestions, and provides information about new products and system applications, both hardware and software.

SECTION 3 - VARIAN 620/f-100 SPECIFICATIONS

Description	System-oriented, general-purpose digital computer for on-line data processing	
Memory	Magnetic core, with a 16-bit word length, 750-nanosecond full-cycle time, 400-nanosecond access time, expandable from the basic 4,096-word (4K) minimum to a maximum of 32,768 words (32K), and asynchronous with CPU operation	
Arithmetic	Parallel, binary, fixed-point, two's complement	
Word Length	16 bits	
Machine Cycle Speed (Fetch and Execute)	Addition/subtraction:	1.5 microseconds
	Multiplication:	6.36 microseconds
	Division:	6.36 Microseconds
	Register modification:	0.75 microseconds
	A/B register input/ output:	3.7 microseconds
	Memory input/output:	4.4 microseconds
Instruction Set	150 instructions, many of which can be microcoded for extended operations	
Instruction Types	One- and two-word addressing, and one- and two-word nonaddressing instructions performing the following functions:	
	Load/store	Jump
	Shift/rotation	Jump and mark
	Register modification	Execution
	Arithmetic	Control
Addressing Modes	Direct, to 2,048 words	
	Relative to P register, to 512 words	
	Preindexed and postindesed with X or B registers, to 32,768 words (does not add to execution time)	
	Multilevel indirect, to 32,768 words	
	Immediate	
	Indirect Indexed, to 32,768 words	
	Extended, to 32,768 words	

VARIAN 620/f-100 SPECIFICATIONS

Operation Registers

A register: 16-bit accumulator and shift register
B register: 16-bit accumulator and shift register (low-order half of the double-length accumulator), or index register
X register: 16-bit index register
P register: 15-bit program counter

Auxiliary Registers

I register: 16-bit instruction register
L register: 15-bit memory address register
R register: 16-bit arithmetic buffer register
D register: 16-bit I/O register

Control Panel

Register entry switches and display indicators; overflow (OVFL), STEP, and RUN indicators; register LOAD and RESET switches; three SENSE switches; instruction REPEAT, and STEP/RUN switches; and three-position power switch

Logic and Signals

Integrated circuits and 18 MHz master clock
Internal logic levels: 0V = false (zero),
+5V = true (one)
Memory data logic levels: 0V = true (one),
+5V = false (zero)
I/O bus logic levels: +3V = false (zero),
0V = true (one)

Input/Output

Programmed I/O operations: external control, program sense, data transfer in, and data transfer out
Automatic data transfers: direct memory access (DMA) with transfer rates up to 275,000 words per second
Interrupt system: allows computer options and peripherals to interrupt CPU operations

Standard Features

Memory protection: protects resident data and programs in memory
Power failure/restart: protects a program in progress during power failure
Hardware multiply/divide: simplifies programming arithmetic operations
Real-time clock: user-selected variable base for time and event accumulation

Computer Options

Teletype controller: provides CPU/Teletype interface logic

VARIAN 620/f-100 SPECIFICATIONS

	Automatic bootstrap loader: saves time and effort involved in manually loading the bootstrap program
I/O Options	Priority interrupt module: establishes and implements interrupt priorities for peripherals Buffer interlace controller: permits direct access to memory for block data transfers Priority memory access: allows up to four external devices direct access to the 620/f-100 memory
Input Voltage	105 to 125V ac, or 210 to 250V ac, at 50 or 60 Hz
Input Current	Power supply requires 8 amperes at 115V, and 4 amperes at 230V
Dimensions	The mainframe is 10.5 inches (26.6 cm) high, 19 inches (48.3 cm) wide, and 21 inches (53.1 cm) deep. The memory chassis is 10.5 inches (25.6 cm) high, 19 inches (48.3 cm) deep, and 19 inches (48.3 cm) wide. An I/O expansion chassis has the same dimensions as the memory chassis. The computer power supply (housed in the mainframe) is 5.2 inches (13.2 cm) high, 18 inches (45.7 cm) deep, and 16 inches (40.6 cm) wide.
Weight	The mainframe weighs 55 pounds (25 kg) without power supply and circuit cards. The memory chassis weighs 25 pounds (10.1 kg) without circuit cards. An I/O expansion chassis weighs the same as the memory chassis. The computer power supply (housed in the mainframe) weighs 46 pounds (20.9 kg) including standard cables.
Temperature	Operating: 0 to 50 degrees C Storage: -20 to 70 degrees C
Humidity	Operating: to 90 percent without condensation Storage: to 95 percent without condensation
Vibration	3 to 10 Hz at 1g force or 0.25 double amplitude, whichever is less; exponentially raised frequency from 3 to 10 Hz and back to 3 Hz for 10 minutes,

VARIAN 620/f-100 SPECIFICATIONS

	three complete cycles; applies to all three principal axes
Shock	4g for 5 to 11 milliseconds, essentially sine shock waveform (all three principal axes, both directions in each axis)
Software	
Symbolic Assembler	Modular two-pass assemblers: 4K version includes 17 basic pseudo-operations; 8K version includes over 30 pseudo-operations; MR macro version operates under the 620 Master Operating System (MOS)
BLD II	Binary load/dump program that allows the loading of object programs and the punching of the binary contents of memory for reloading
Subroutines	Complete library of basic mathematical and utility subroutines; provides for the addition of special-purpose routines
FORTTRAN	Modular one-pass compiler; subset of ANSI FORTRAN for 8K of memory
BASIC	An easy-to-use programming language for business and scientific applications that permits an inexperienced operator to program the system with only a few hours training
MOS	A master operating system that provides for automatic batch-processing in as little as 8K of memory
RPG IV	An optional report program generator system for business applications; produces reports, financial statements, sales records, and other commercial documents
AID II	Program analysis package that assists programmers in operating the computer and debugging other programs; includes basic operational executive subroutines

EDIT	Program modification package for on-line correction of paper tape
MAINTAIN II	Software package that provides efficient off-line verification of CPU and peripheral operation and assists in isolating and correcting suspected faults
VORTEX	A multiprogramming operating system that provides the versatility offered by large, expensive computer systems; VORTEX permits concurrent real-time programming and background processing

SECTION 4 - CENTRAL PROCESSING UNIT

The **Varian 620/f-100 Computer** is organized in three major functional sections:

- a. The Central Processing Unit (CPU)
- b. The Memory
- c. The Input/Output Section

The memory is described in detail in section 5, and the external I/O structure in sections 7 and 12. The CPU, memory, and I/O interface, and standard features are described in the following paragraphs. A block diagram of the Varian 620/f-100 computer is shown in figure 4-1.

CPU Section

The CPU section consists of clock circuits, data buses, operation and auxiliary registers, arithmetic unit, control and decoding circuits, and register display and switch circuits.

The bit slice technique is used in structuring the arithmetic unit, control and decoding circuits, and register display and switch circuits.

Clock Circuits

Two crystal-controlled clock circuits on the clock card provide basic timing signals. The main clock circuit generates an 18 MHz master clock to the CPU control circuits for memory interface. The main clock circuit also divides the 18 MHz clock by two and applies the resulting 9 MHz clock to the control circuits for state control.

The I/O clock circuit generates an 8.8 MHz master clock for the I/O section.

Buses

The basic computer contains eight buses designated as C, AY, AZ, MB, MD, ABS, AB, and I/O.

The C bus provides the data path and selection logic for routing data from the arithmetic unit to the operation registers (A, B, X, and P), the auxiliary registers (D, I, and L), and register display.

The AY bus routes selected data from the I and R registers and register entry switches.

The MB and MD buses provide data paths to and from memory, respectively.

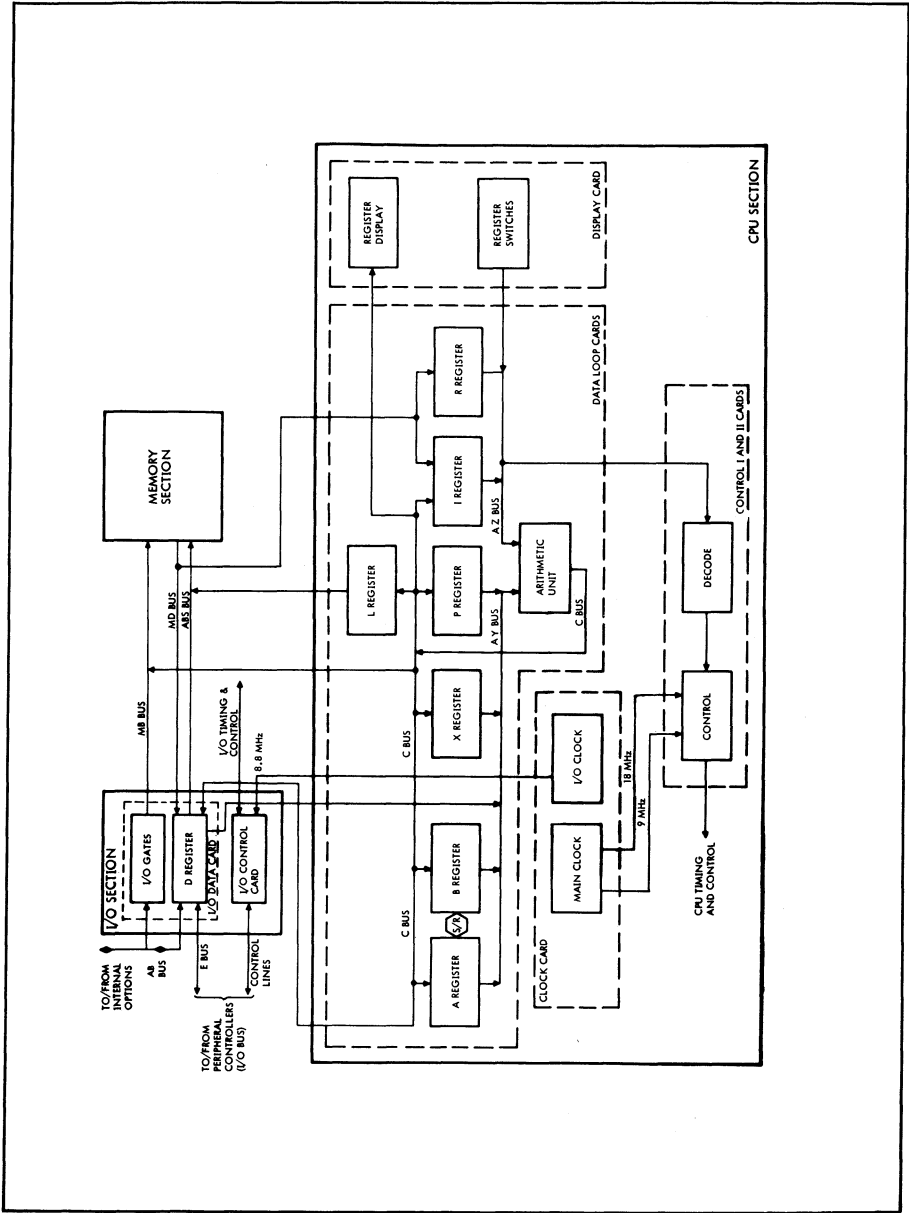
The ABS bus routes address information into memory from the address (L) register and the D register.

The AB bus provides data paths between the computer and the RTC, PF/R, MP, and Teletype controller.

Operation Registers

The A, B, X, and P registers comprise the Varian 620/f-100 operation registers. The A, B, and X registers can be directly accessed. The P register is indirectly accessed through use of the following instruction types:

- a. Jump instructions that modify the program sequence.
- b. Arithmetic/logic instructions in the relative-addressing mode that use



VT12-0281B

Figure 4-1. Varian 620/f-100 Block Diagram

the contents of the P register to modify the operand address.

The outputs of the operation registers are connected to the AY bus and routed to the AY inputs of the arithmetic unit. The inputs to the operation registers come from the C bus, which contains output data from the arithmetic unit.

The A register is a 16-bit register that serves as the upper half of the accumulator. It is used for:

- a. Accumulating the result of logical operations
- b. Accumulating the result of addition or subtraction
- c. Accumulating the more significant half of the product in multiplication (optional)
- d. Accumulating the remainder in division (optional)
- e. Program-controlled I/O transfers

A shift/rotate (S/R) circuit provides a special data path to implement A and B register shift and rotate operations.

The B register is a 16-bit register that serves as the lower half of the accumulator. It is used for:

- a. Accumulating the less significant half of the product in multiplication
- b. Accumulating the quotient in division
- c. Program-controlled I/O transfers
- d. Indexing (as a second index register)

The X register is a 16-bit register that permits the indexing of operand addresses without adding time to the execution of indexed instructions.

The P register is a 15-bit register that contains the address of the current instruction. It is incremented before each new instruction is fetched.

Auxiliary Registers

The I, R, L, and D registers comprise the auxiliary registers. The D register is in the I/O section. The outputs of the I and R registers are connected to the AZ bus and routed to the AZ input of the arithmetic unit. The inputs to the I and R registers come from the MD bus, which contains memory output data.

The I register is a 16-bit register that receives each instruction from memory through the MD bus and holds the instruction during its execution. Instructions can be loaded in the I register from the C bus via the control panel register entry switches. The control fields of the instruction word are routed to the decoding section to determine the required timing and control signals. The five least significant bits of the I register are transferred into a shift counter to shift-count-control the shift instructions.

The R register is a 16-bit buffer register that buffers the arithmetic unit from memory to permit interlaced I/O operations to steal memory cycles. The R register holds the second word of a two-word instruction, the multiplicand in multiplication, and the divisor in division.

The L register is a 15-bit address register that contains the address of the memory location currently being accessed during either a clear/write or read/restore cycle.

CENTRAL PROCESSING UNIT

Arithmetic Unit

The arithmetic unit contains the adder and gating for all operations except shifting. Under the control of the CPU control circuits, the arithmetic unit processes data from the AY and AZ buses and sets the result on the C bus.

Control Circuits

The control circuitry generates timing and control signals using the information received from the I register. The control circuits are contained on the control I and II cards.

Decoding Circuits

The decoding circuitry decodes the bits of the word in the I register used to determine control signal levels generated by the control circuits. The decoding circuits are contained on the control I and II cards.

Register Display and Switch Circuits

This section displays the contents of and loads registers or memory on the Varian 620/f-100 control panel. The register display and switch circuits are contained on the display card.

Memory Interface

The expandable, random-access, three-wire/three-dimensional magnetic core memory is used for program and data storage. It operates asynchronously with the CPU and I/O sections.

Instructions from memory are transferred to the control section for execution. Under program control, words can be transferred from memory to the arithmetic unit, opera-

tion registers, or I/O bus, and conversely, to memory from the operation registers or the I/O bus.

Words written into memory are transferred on the MB bus; words read out, on the MD bus. Memory addresses are transferred to memory on the unidirectional ABS bus.

A modular master/slave configuration permits economical memory expansion. The slave module (or modules) uses the timing and control circuits of the master memory module.

I/O Interface

Peripheral devices communicate with the computer on the I/O bus. The 16-bit, party-line, bidirectional I/O bus transfers programmed data between peripheral devices and the computer. The I/O bus also permits plug-in expansion of Varian 620/f-100 peripherals. Part of the I/O bus is an E bus for bidirectional data transfer.

Information exchanges with peripheral devices are synchronized by peripheral controllers, which can control one or more similar devices. Each controller and the device (or devices) it controls comprise a peripheral device option.

The I/O section is on two circuit cards. The I/O data card contains the 16-bit D register, which stores E bus I/O information. The I/O control card contains timing and control circuits for program-controlled I/O, cycle-stealing I/O, and interrupt operations.

Standard Features

The following standard features are included in the mainframe chassis of each Varian 620/f-100 computer:

- a. Memory protection
- b. Power failure/restart
- c. Hardware multiply/divide
- d. Real-time clock

Memory Protection

The MP implements the designation of protected and unprotected areas of memory and prevents unauthorized entry into and modification of protected areas by programs operating in unprotected areas.

Functional Description

The MP divides computer memory into blocks (or segments) of 512 consecutive words. Under program control, these segments are selectively designated protected. Segments not specifically designated protected are, therefore, unprotected. The MP stores the protected/unprotected status of the segments in four 16-bit mask registers. These registers can store the status of up to 64 memory segments, i.e., up to 32K words.

Using the stored memory segment status, the MP monitors the address of the current instruction and that of the following instruction and the memory location specified by the effective address to determine error conditions.

When a program is operating from an unprotected segment, the MP detects the following operations as errors:

- a. Program overflow into a protected area
- b. Writing in a protected area
- c. Jumping to a protected area
- d. Executing an I/O instruction in an unprotected area

- e. Executing a halt instruction in an unprotected area

If these operations are attempted, the program aborts and jumps to a preassigned address in memory, from which the computer is directed to a user-written service subroutine.

The MP specifications are listed in table 4-1.

Power Failure/Restart

The PF/R provides an orderly shutdown in case of power failure or turn-off and restarts the interrupted program when power is restored.

Functional Description

Power input to the computer is indirectly monitored by the PF/R. A power failure monitor voltage in the computer power supply is constantly being sensed to determine power status. If the voltage drops (because of power failure or turn-off), the PF/R initiates an interrupt. The CPU then executes a user-written service subroutine (SAVE) that places the contents of the volatile registers (A, B, X, P, and overflow) in memory. Execution of the stored program stops, memory is disabled, and the system is reset. This power-down subroutine cannot be interrupted by lower-priority options or peripheral controllers.

When power is restored, the PF/R enables the memory. The CPU executes a user-written service subroutine (RESTORE) that reloads the contents of the volatile registers, and the system resumes execution of the program in progress at the time of the interrupt.

If power fails while the computer is in step mode, memory is protected, but the contents of the volatile registers are lost.

The PF/R specifications are listed in table 4-2.

Table 4-1. MP Specifications

Parameter	Description
Organization	Consists of A-bus receiver, device address decoder, function control logic, segment address register, mask register, segment address status logic, error detector, CPU control logic, interrupt request logic, instruction address register, and A-bus driver
Capability	Protects up to sixty-four 512-word memory segments
Operational Modes	Detects halt, overflow, I/O, write, and jump errors
Priority Assignment	Always assigned the highest priority with respect to the DMA/interrupt scheme
I/O Capability	Six external control and eight data transfer instructions
Timing Sources	18 MHz and 9 MHz clocks from CPU
Logic Levels	
Internal	Positive logic: True: +2.4 to +5.25V dc False: 0.0 to +0.45V dc
I/O	Negative logic: True: 0.0 to +0.45V dc False: +2.4 to +5.25V dc
Size	One 3-by-15-inch (7.7 x 38.1 cm) wired-socket circuit card
Interconnection	Plugs into slot 12 of the 620/f-100 CPU tray
Input Power	+5.0V dc at 2.5 amperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 4-2 PF/R Specifications

Parameter	Description
Organization	Contains control sequencer logic, system start logic, interrupt request logic, power-up/power-down logic, and interrupt priority logic
Interrupt Memory Address	SAVE routine at 040 and 041 and RESTORE routine at 042 and 043
Priority Assignment	Normally second-highest with respect to DMA/ interrupt priority assignment (exceeded only by MP)
Power-Down Timing	Disables memory and the CPU in less than 2 milliseconds after power loss; up to 300 microseconds allowed for SAVE routine execution
Power-Up Timing	Power-up restoration is typically 40 milliseconds; up to 400 microseconds allowed for RESTORE routine to be executed
Logic Levels	
I/O	Negative logic:
	True: 0.0 to +0.5V dc
	False: +2.8 to 3.6V dc
Internal	Positive logic:
	True: +2.4 to +5.5V dc
	False: 0.0 to +0.5V dc
Size	One 3-by-15-inch (7.7 x 38.1 cm) circuit card shares card with RTC
Interconnection	Plugs into 620/f-100 CPU tray
Input Power	+5V dc $\pm$ 5 percent at 0.5 ampere
Operational Environment	0 to 50 degrees C, 0 to 90 percent relative humidity without condensation

CENTRAL PROCESSING UNIT

Hardware Multiply/Divide

The M/D enhances the computational capabilities of the Varian 620/f-100 computer by reducing the number of steps required for multiplication and division operations.

During multiplication, the contents of the effective memory address are multiplied by the contents of the B register, then the A register contents are added to the product. The result is placed in the A and B registers, with the most significant half in the A register and the least significant half in the B register. The sign of the result is in bit 15 (sign bit) of the A register. The B register sign bit is always set to zero. The largest positive multiplier or multiplicand in an operation register in the CPU is thus 15 binary bits. Execution time is 6.75 microseconds, or 9 machine cycles.

During division, the dividend is contained in the combined A and B registers with the sign in bit 15 of the A register. The B register sign bit is not used. The divisor is in the effective memory address. The quotient and its sign are placed in the B register and the remainder (with the sign of the dividend), in the A register. If the quotient exceeds the capacity of the B register (+32,767 or -32,678), the over-

flow indicator sets. Execution time is 6.75 microseconds, or 9 machine cycles.

M/D specifications are listed in table 4-3.

Real-Time Clock

The RTC provides flexible time orientation that can be used for time-of-day and event accumulation and as an interval timer. Known periods of time are generated by the RTC for program-sequencing. A free-running counter that can be read under program control is also provided.

The main program is periodically interrupted by the RTC to initiate subroutines which accomplish the desired real-time functions. The RTC can generate two types of interrupts: variable interval and memory overflow. The first interrupt is a time-base signal that increments a specific memory address when recognized by the computer. The second interrupt occurs when the incremented memory address reaches a count of 040001. The frequency of the increment interrupt is adjustable with program-controlled I/O and can range from 3 to 10,000 per second. A predelivery factory adjustment sets the RTC to initiate one increment interrupt every millisecond unless otherwise programmed. An externally generated time base can be used.

RTC specifications are listed in table 4-4.

Table 4-3. M/D Specifications

Parameter	Description
Organization	Contains multiply/divide and shift control logic, and extended addressing control logic
Multiplication Algorithm	$R \cdot B + A = A, B$ where A = initial A register contents B = multiplier in B register R = multiplicand in memory A,B = result in the A and B registers (most significant half in A, least significant half in B) The A register sign bit contains the sign of the product; the B register sign bit is reset to zero
Multiplication Capability	Maximum multiplier: 32,767 (- 32,768) Maximum multiplicand: 32,767 (- 32,768) Maximum product: 1,073,741,824 (exceeds the capacity of the A and B registers by one and sets overflow)
Division Algorithm	$A, B / R = B, A$ where A,B = dividend in the A and B registers (most significant half in A, least significant half in B) R = divisor in memory B,A = result (quotient in the B register, remainder in the A register) The B register sign bit contains the sign of the quotient; the A register sign bit, the sign of the dividend
Division Capability	Maximum divisor: 32,767 (- 32,768) Maximum dividend: 1,073,741,823 (- 1,073,741,824) Maximum quotient: 32,767 (- 32,768) A larger quotient sets the overflow indicator

Table 4-3. M/D Specifications continued

Parameter	Description
Logic Levels	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc Negative (interface) logic: True (low): 0.0 to +0.5V dc False (high): +2.4 to +5.5V dc
Size	One 3-by-15-inch (7.7 x 38.1 cm) wired-socket card
Input Power	+5V dc at 0.6 ampere
Interconnection	Plugs into the 620/f-100 CPU tray
Operational Environment	0 to 50 degrees C, 10 to 90 percent relative humidity without condensation

Table 4-4. RTC Specifications

Parameter	Description
Organization	Contains instruction and decode logic, divide-by-110 logic, free-running counter, input timing source, variable-interval logic, memory overflow detect logic, and interrupt control logic.
Priority Assignment	Normally third-highest with respect to DMA/ interrupt priority assignment (exceeded only by MP and PF/R
I/O Capability	Seven external control and eight transfer instructions
Operating Modes	Variable-interval interrupt, memory overflow interrupt, interval accumulation, time-of-day accumulation, and event accumulation

Table 4-4. RTC Specifications(continued)

Parameter	Description
Timing Options	Three hardware-selected timing sources for the variable-interval interrupt and three hardware-selected timing sources for the free-running counter
Timing Sources:	
10-kHz	± 0.1 percent, squarewave; variable interrupt range, 100 microseconds to 409.5 milliseconds (in 100-microsecond increments)
Line Frequency	24V rms, sine wave; variable interrupt range (nominal), 16.7 milliseconds to 68.3 seconds at 60 Hz (in 16.7-millisecond increments) and 20.0 milliseconds to 81.9 seconds at 50 Hz (in 20.0-millisecond increments)
External (User-Supplied)	5.0 microseconds minimum positive and minimum negative duration
Free-Running Counter Capacity	0177777
Logic Levels	
I/O	Negative logic: True: - 0.5 to + 0.5V dc False: + 2.4 to + 5.0V dc
Internal	Positive logic: True: + 2.4 to + 5.0V dc False: - 0.5 to + 0.5V dc
Size	One 3-by-15-inch (7.7 x 38.1 cm) wired-socket card; shares card with PF/R
Interconnection	Plugs into the 620/f-100 CPU tray
Input Power	+ 5V dc ± 5 percent at 2.0 amperes
Operational Environment	0 to 50 degrees C, 0 to 90 percent relative humidity without condensation

SECTION 5 - MEMORY

The Varian 620/f-100 memory is a parallel, random-access, three-wire/three-dimensional, magnetic-core memory. It provides program and data storage for the Varian 620/f-100 computer. The basic computer system package accommodates 4,096 (4K) or 8,192 (8K) sixteen-bit words. The memory can be expanded to 32,768 (32K) words in 4K increments.

Memory Design

The basic information storage element of the memory is the magnetic core. The core is a toroid of ferrite that can be magnetized in two discrete directions representing a binary one or zero. The core is magnetized by a current passing through it. The direction of current flow through the core determines the direction of magnetization, either read or write.

The Varian 620/f-100 memory uses the coincident-current technique for core magnetization. Two perpendicular wires (X drive and Y drive) pass through each core. During memory operations, the current on each drive wire is approximately one-half the current necessary to magnetize a core; thus, only the core at the intersection of two activated drive wires is magnetized.

Memory Operations

A memory full-cycle consists of a reading sequence followed by a writing sequence. The full-cycles are:

- b. **Read/restore:** a word is transferred from memory to the CPU

Both full-cycle operations require 750 nanoseconds and memory access time is 400 nanoseconds.

A clear/write operation loads the memory. The clear half-cycle of the operation resets the addressed cores to zero; the write sequence then loads the data in the cores.

A read/restore operation reads information from the memory. The read half-cycle of the operation unloads the addressed data word from memory; the restore sequence immediately reloads (writes the word in the same location).

Circuit Cards

The Varian 620/f-100 memory circuits are on four types of circuit cards:

- a. *The memory stack card* contains a diode-decoding matrix and a 4K-by-16 magnetic core array. Each 4K memory increment requires one stack card.
- b. *The driver/sink switch card (part number 44P0578)* contains 16 driver and 16 sink switch pairs, address multiplexer and decoder, driver/sink switch timing, and current source circuits. Each 8K memory increment requires one driver/sink switch card.
- c. *The sense/inhibit card (part number 44P0506)* contains 16 sense ampli-

MEMORY

fiers and 16 inhibit drivers. Each 4K memory increment requires one sense/inhibit card.

- d. *The memory timing and register card (part number 44P0579)* contains timing control circuits, address register, stack-select decoder, data register, and line drivers. Only one memory timing and register card is required for any size memory system.

Figure 5-1 illustrates the functional circuit blocks of the memory in relation to the four circuit cards.

Interfacing And Timing

The memory is asynchronous with the CPU in a request-response mode. The CPU requests memory by generating a memory start pulse; the memory responds by generating a data ready pulse. Memory interface with the CPU and I/O sections is through three unidirectional buses: address information is transferred to memory on the ABS bus, data are read from memory on the MD bus, and data are written into memory on the MB bus.

Figure 5-2 shows the memory interface waveforms. The CPU sends a low Memory Start (MST) signal to address memory. The memory cycle begins after a settling-time delay of no more than 50 nanoseconds. The write portion of the Read/Write (RW -) signal strobes write data (MB00-MB15) into the memory data register prior to the leading edge of the Data Ready (MDR -) pulse. MDR - is generated within 400 nanoseconds after the memory cycle begins, indicating that memory has accepted write data from the MB bus, or has transferred read data (MD00 - MD15) to the CPU. The trailing edge of MDR -

clears MST - memory address (ABS00 through ABS14), and write data at the CPU.

Wrap-Around Addressing

Wrap-around addressing is available (upon request) as a standard feature of the Varian 620/f-100 memory. Wrap-around addressing techniques automatically prevent the CPU from trying to address data to or from nonexistent memory addressing. Without this capability, data read out of a nonexistent address result in zero, and data written are lost.

Wrap-around addressing is accomplished by inhibiting (holding at a high logic level) one or more of the three most significant address bits (ABS12 -, ABS13 -, and ABS14 -). This is implemented by adding jumper wires between slots 1 and 2 of the memory chassis backplane.

In a computer with a 4K memory (symbolically, addresses 0 through 4,095) wired for wrap-around addressing, each existent memory location can be addressed directly and by seven corresponding addresses in nonexistent memory; e.g., address 0 can be addressed by 0, by 4,096, by 8,192, by 12,288, by 16,384, by 20,480, by 24,576, and by 28,672. For 8K wrap-around, each existent location can be addressed by its own designation plus three nonexistent addresses, and for 16K, its own plus one nonexistent address.

Specifications

Performance specifications for the Varian 620/f-100 memory system are listed in table 5-1.

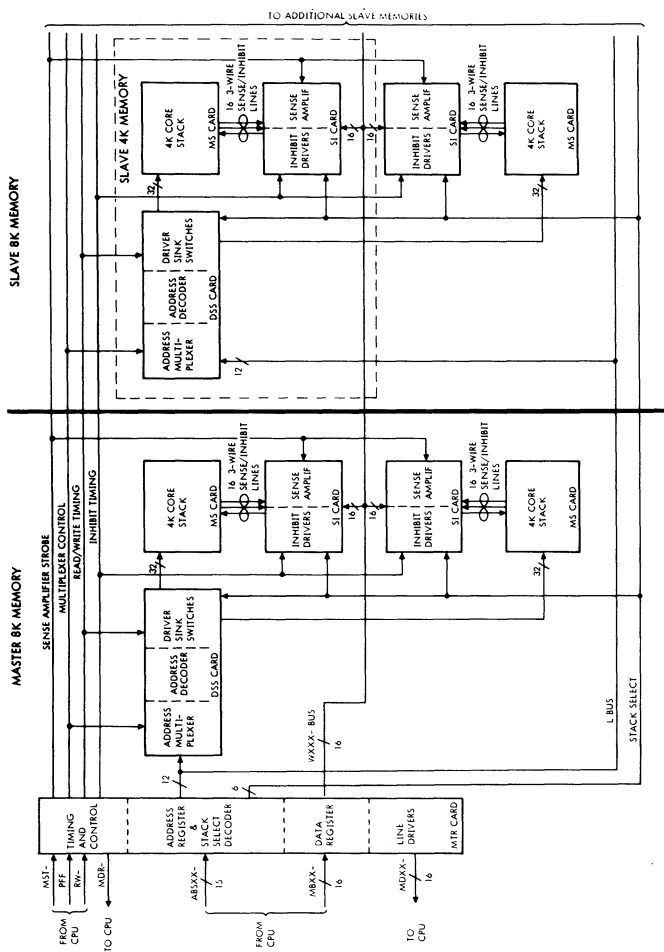
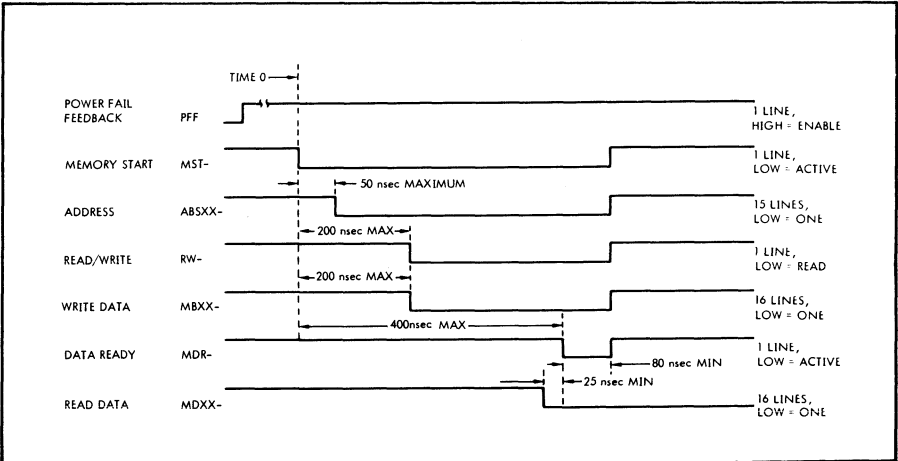


Figure 5-1. Varian 620/f-100 Memory Block Diagram

MEMORY

Table 5-1. Memory Specifications

Parameter	Description
Description	Parallel, random-access, 3W/3D, magnetic core memory
Cycle Time	750 nanoseconds
Access Time	400 nanoseconds
Execution Times	Write data available from 150 to 350 nanoseconds; address available at 50 nanoseconds (settled); memory start pulsewidth, 80 nanoseconds (minimum)
Operating Modes	Continuous at any address, and in burst mode (i.e., memory cycle series interspersed by nonoperating time)
Operating Limits	X/Y current: ± 5 percent (minimum) Inhibiting current: ± 5 percent (minimum) Power supply voltages: can be biased ± 5 percent from nominal
Operating Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation



VTII-1340

Figure 5-2. Memory Interface Waveforms

SECTION 6 - COMPUTER OPTIONS

Three mainframe computer options are available for the Varian 620/f-100:

- a. Teletype Controller (TC)
- b. Automatic Bootstrap Loader (ABL)
- c. Priority Memory Access (PMA)

All three computer option circuit cards can be installed in the mainframe. The PMA option is considered an I/O option, and is described in section 8.

Teletype Controller

The peripheral controller for the first Teletype (Models 620/f-06, -07, -08) in the Varian 620/f-100 system is assigned a slot in the mainframe (slot 13). It interfaces with the CPU through the backplane wiring; refer to section 11 for interconnection details.

If more than one Teletype is required, additional peripheral controllers can be installed in any I/O slot in the memory or expansion chassis. The additional Teletypes also require the addition of the Teletype buffer board (model 620-09). As many as eight Teletypes can be included in one 620/f-100 system.

The factory-modified Teletype unit controlled by the Teletype controller can be a Teletype Model 33 ASR, 35 ASR, or 35 KSR. The ASR models include paper-tape reader and punch capabilities; the KSR model uses only keyboard-entered instructions and data.

Specifications of the Teletype controller are given in table 6-1. Refer also to the Teletype controller manual (document number 98 A 9908 160).

Table 6-1. Teletype Controller Specifications

Parameter	Description
Organization	Contains input and output registers, timing control circuitry for simultaneous two-way communication, and CPU/Teletype interface logic
Control Capability	One factory-modified Teletype Model 33 ASR, 35 ASR, or 35 KSR, including cable
I/O Capability	Three external control, eight transfer, and two program sense instructions

Table 6-1. Teletype Controller Specifications (continued)

Parameter	Description
Operation Modes	Input: from keyboard or paper-tape reader Output: to Teletype printer or paper-tape punch
Interrupt Capability	Ready to write and ready to read interrupt available to PIM
Standard Device Address	01 through 07
Memory Access Control	By CPU indirectly (via Teletype buffer board), or by BIC
Logic Levels	Positive Logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc
Size	First controller is contained on one 3-by-15-inch (7.7 x 38.1 cm) wire-wrapped circuit card. additional controllers are contained on 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit cards
Interconnection	Connects to Teletype unit via a cable that connects to rear of memory chassis
Power Consumption	+5V dc at 1.2 amperes
Operational Environment	0 to 50 degrees C; 10 to 90 percent relative humidity without condensation

Automatic Bootstrap Loader

The automatic bootstrap loader (ABL), model 620/f-115, is a hardware option that automatically loads the 620/f-100 bootstrap program into core memory from

an integrated-circuit ROM. This option saves the time and effort involved in manually loading the bootstrap program.

Specifications of the ABL are given in table 6-2.

Table 6-2. ABL Specifications

Parameter	Description
Organization	Consists of an IC ROM address counter, control logic, and BOOTSTRAP switch on the control panel
ROM Program	Standard ABL option provided with the ROM programmed for the high-speed paper tape reader or Teletype. As a special option, the standard ROM can be replaced with one specially programmed for any other I/O device if the bootstrap program does not exceed thirty-two 16-bit words
Data Transfer	16-bit transfers from the ROM to the 620/f-100 memory via the DMA channel; hardware insertion of instructions in the P, I, and X registers
Options	Specially programmed ROMs for various I/O devices, card reader, drum, disc, magnetic tape units, etc.
Logic Levels	Positive Logic True: +2.5 to +5.0V dc False: 0.0 to +0.5V dc
Size	One 3-by-15-inch (7.7 x 38.1 cm) wire-wrapped circuit card; shares card with first Teletype controller
Interconnection	Plugs into 620/f CPU tray
Power Consumption	+5V dc at 900 Milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

SECTION 7 - INPUT/OUTPUT SYSTEM

The Varian input/output (I/O) system allows the computer to interface with a large variety of peripheral devices. Interface circuitry to control a specific peripheral is contained on one or more controller cards that plug into a peripheral controller slot in the mainframe or expansion chassis. One controller can control one or more similar peripherals.

The I/O system comprises the following principal elements:

- Timing and control logic and internal bus interface in the CPU
- Various peripheral controllers
- A party-line, time-shared I/O bus connecting the CPU and controllers
- I/O options that enhance the I/O capabilities

The CPU is described in sections 4 and 30. Information on the controllers for specific peripherals is given in section 9. The I/O options are briefly described below, with details of each outlined in section 8. I/O instructions are described in detail in section 16. This section concentrates on the role of the I/O bus and the interaction between the I/O bus and system peripheral controllers.

I/O Options

Three I/O options are available with the Varian 620/f-100 computer.

The Priority Interrupt Module (PIM) allows peripheral control-

lers in the system to interrupt CPU operations to initiate the servicing of interrupt subroutines.

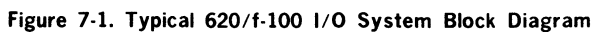
The Buffer Interface Controller (BIC) permits peripherals to transfer data directly to or from memory at rates up to 275,000 words per second by implementing a cycle-stealing, direct memory access (DMA) trapping technique.

The Priority Memory Access (PMA) allows up to four external devices direct access to the Varian 620/f-100 memory. The PMA interfaces with PMA controllers, which are either designed by the user or supplied by Varian.

The standard Varian 620/f-100 contains all the hardware necessary to implement the PIM and BIC options. The following discussion assumes that both are included in the I/O system.

Organization

The I/O system utilizes a bidirectional I/O bus, which allows one set of data and control lines to communicate with all system peripherals. The organization of the I/O system is illustrated in figure 7-1. All peripheral controllers and I/O options connect to the I/O bus (excluding the first system Teletype controller, which plugs directly into the 620/f-100 CPU tray). The I/O bus cable plugs into the backplanes of the mainframe and expansion chassis.



The priority chain, BIC control, interrupt request, trap request, and acknowledgment lines shown in figure 7-1 are contained in the I/O bus; they are separated in the illustration for clarity. However, the interrupt lines to the PIM are separate entities, installed as needed during system installation or expansion.

I/O Bus Structure

The Varian 620/f-100 communicates directly with peripherals by transmitting an external control instruction and peripheral device address to the selected controller via the I/O bus. When a peripheral is ready to send or receive information, as indicated by its associated sense line, the computer requests the device to place a word of data, or to accept one placed by the computer, on the I/O bus. The I/O bus

lines are described in the following subsections. System interconnection details are given in section 11.

E Bus

This 16-bit, parallel, bidirectional I/O channel (EB00 - I through EB15 - I) is used to transmit external control instructions, device addresses, and data from the computer to the peripherals. In turn, the E bus is used by the peripherals to transmit data to the computer. Ten drivers and ten receivers can be connected to each line to service up to ten peripheral controllers. When more than ten controllers are included in the system, they are controlled through an I/O party-line expander card. An E bus signal is true when it is at 0V dc, and false at +3V dc. Figure 7-2 shows a typical E bus configuration.

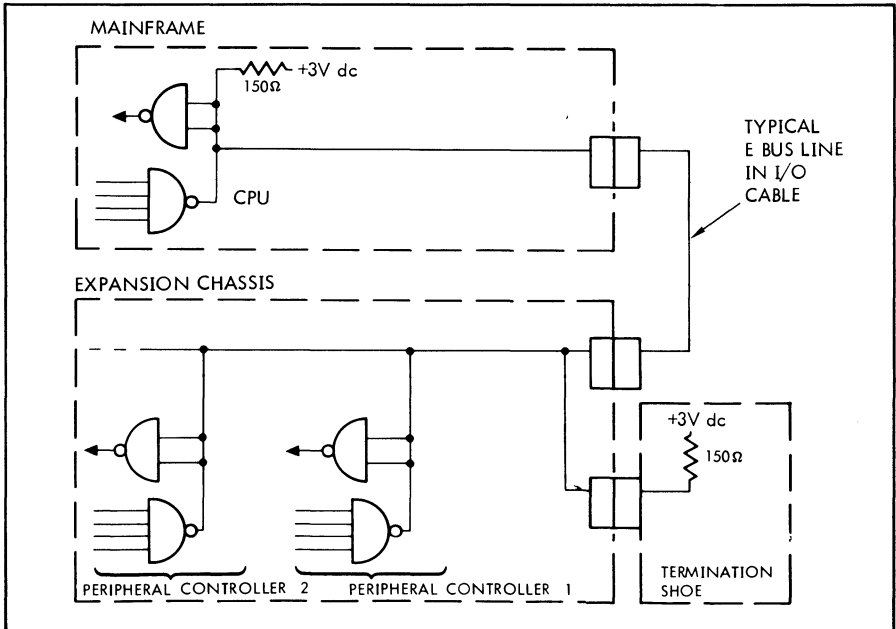


Figure 7-2. Typical E Bus Line Configuration

INPUT/OUTPUT SYSTEM

Control Lines

FRYX-I: This signal is generated by the computer to indicate that an external control instruction and a device address have been placed on the E bus. Each peripheral controller examines the device address, and, upon the true-to-false transition of FRYX-I, the addressed peripheral responds to the external control instruction. FRYX-I is true at 0V dc and false at +3V dc.

DRYX-I: This signal indicates that the computer has placed data on the E bus, or that it has accepted the data placed on the E bus by the peripheral. The transfer occurs upon the true-to-false transition of DRYX-I. DRYX-I is true at 0V dc and false at +3V dc.

IUAX-I: This signal is generated by the computer to acknowledge the receipt of an interrupt, trap-in, or trap-out request. The interrupting or trapping peripheral controller can communicate an address to the computer and can receive data from or send data to the computer when IUAX-I is true. IUAX-I also inhibits device address decoding in all controllers during the address phase of an interrupt or trap operation to prevent the controllers from interpreting any part of the memory address as a device address.

SYRT-I: This signal is used to initialize all the peripheral controllers connected to the I/O bus. SYRT-I is true when the RESET switch on the control panel is pressed. SYRT-I is true at 0V dc and false at +3V dc.

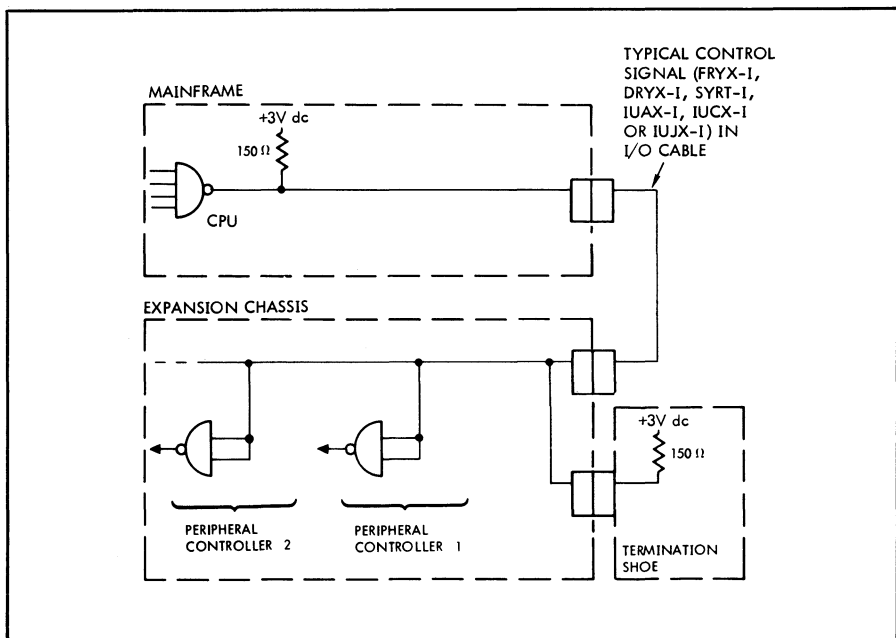


Figure 7-3. Typical Control Line From the Computer

SERX-I: This signal is a controller response to a program sense instruction, during the execution of which the computer places a function code and a device address on the E bus. The addressed controller is instructed to indicate the status of a peripheral device action. If the action can be performed, the controller responds by setting SERX-I true.

IUCX-I: This signal is an interrupt-synchronization clock from the computer that is disabled when IUAX-I is true. The true-to-false transition of IUCX-I sets the interrupt request flip-flops in the appropriate peripheral controllers.

IURX-I: An interrupting peripheral controller requests the computer to execute an instruction by setting this signal true. The address of the instruction is placed on the E bus when the computer acknowledges the interrupt request (IUAX-I).

IUJX-I: This signal is generated by the computer to inhibit all interrupts following a jump-and-mark instruction when that instruction is the result of an interrupt request. IUJX-I becomes true 2.7 microseconds from the false-to-true transition of IUAX-I and remains true for 450 nanoseconds.

TPIX-I: The BIC sets TPIX-I true to request the computer to input one word of data to the memory. The address is placed on the E bus by the controller when the computer acknowledges the request.

TPOX-I: The BIC sets TPOX-I true to request the computer to output one word of data from memory. The address is placed on the E bus by the controller when the computer acknowledges the request.

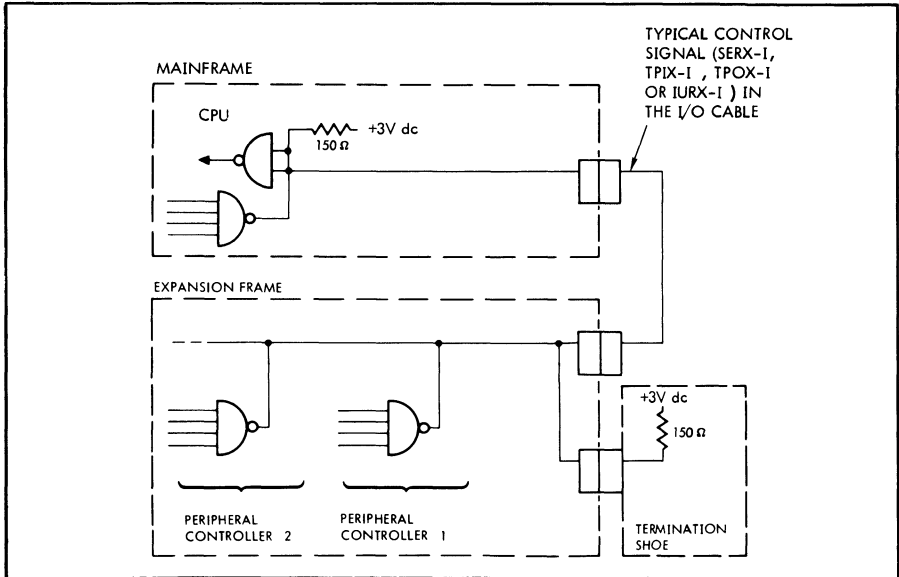


Figure 7-4. Typical Control Line to the Computer

INPUT/OUTPUT SYSTEM

Priority Lines

PR1X-I, PR2X-I, PR3X-I, and PR4X-I are used to establish the priority of system interrupts. They connect devices in a priority chain. The devices that can be included in this priority chain (in order from high to low priority) are: MP, PF/R, RTC, PIM, BIC, and interrupt (INT) switch on the control panel. Peripheral controllers cannot, of themselves, generate interrupt requests; they do so only through interaction with the PIM. System interrupts are discussed in greater detail later in this section.

BIC Control Lines

These seven control lines are used for communication between a BIC and the peripheral controllers it monitors.

I/O Operations

In the Varian 620/f-100 I/O system, information transfers can occur under the control of a stored program, they can be interrupt-initiated, or they can occur on a cycle-stealing, trapping basis. Specific details of these operations are contained in the 620/f-100 Maintenance Manual (document 98 A 9908 150). The following paragraphs briefly outline the capabilities of the system.

Program-Controlled I/O

The I/O system provides four types of I/O operations under program control:

- a. **External Control.** An external control code, specifying a specific peripheral function and a device address, is transmitted from the computer to a peripheral controller.

- b. **Program Sense.** The status of a selected peripheral controller sense line is interrogated by the computer.

- c. **Input Data Transfer.** One word of data is transferred from a peripheral controller to the A register, B register, or a location in memory.

- d. **Output Data Transfer.** One word of data is transferred to a peripheral controller from the A register, B register, or a location in memory.

The instructions implementing these operations are described in section 16.

Under program control, the I/O system communicates directly with all peripherals. The computer can initiate peripheral operations by transmitting an external control function code and a proper device address to the selected controller via the I/O bus. The computer can determine when a peripheral is ready to send or receive information by interrogating its associated sense line. A peripheral can be requested to place a word of data on the I/O bus during a computer input transfer, or to accept a word of data placed on the bus by the computer during an output transfer.

Table 7-1 summarizes how the E bus and control signals determine transfers on the I/O bus during I/O operations. Figures 7-5 and 7-8 show the timing for the program control operations.

An I/O instruction is not transmitted intact over the E bus. The function code (bits 0-8) of the instruction are transmitted unchanged. Bits 9-15 are decoded in the CPU to generate the configuration of EB11-I through EB15-I required by the specified operation onto the bus.

Table 7-1. E Bus and I/O Control Signals

OPERATION > CONTROL LINE >	External Control	Sense	Data Transfer		Trapping Sequence	Interrupt Sequence
			FRYX-I* (Phase 1)	DRYX-I (Phase 2)		
EB00-I		FRYX-I* SERX-I* (Phase 1)			TPOX-I or TPIX-I IUAX-I, FRYX-I (Phase 1)	IURX-I IUAX-I (Phase 1)
EB01-I						
EB02-I	Device address	Device address	Device address			
EB03-I						
EB04-I						
EB05-I						
EB06-I	Function code	Function code	Not used	Data	Address	Data
EB07-I						
EB08-I						
EB09-I	Not used	Not used				
EB10-I						

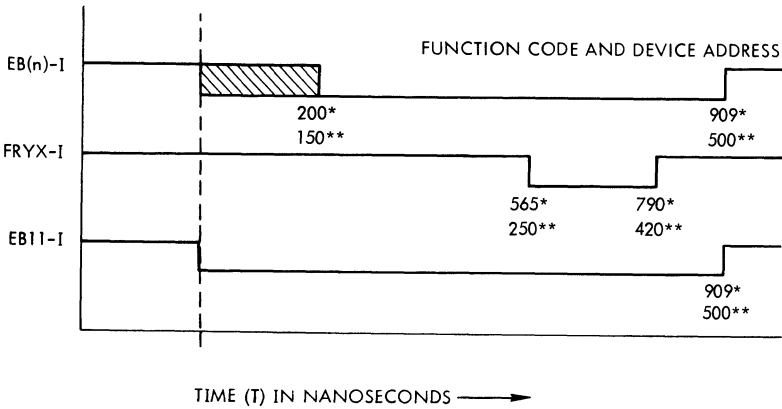
Pairs of
signals
used for
specific
interrupts

Table 7-1. E Bus and I/O Control Signals continued

OPERATION >	External Control	Sense	Data Transfer	Trapping Sequence	Interrupt Sequence
CONTROL LINE >	FRYX-I* (Phase 1)	FRYX-I* SEPX-I* (Phase 1)	FRYX-I* (Phase 1) DRYX-I (Phase 2)	TPOX-I or TPIX-I IUAX-I, FRYX-I (Phase 1) DRYX-I (Phase 2)	IURX-I IUAX-I (Phase 1)
EB11-I	External control command	Zero	Zeros	Address	Pairs of signals used for specific interrupts
EB12-I	Zeros	Sense command			
EB13-I		Zeros	Data in		
EB14-I			Data out		
EB15-I	See note 3		Zero	Data	

- NOTES:
1. Phase 1 is device or memory selection.
 2. Phase 2 is the data transmission.
 3. For extended external control, control and data lines are the same as external control except EB11-I is zero and EB15-I is one.

* IUAX interlock; used in address decoding.



T = 0 is the start of the execute phase of the external control instruction.

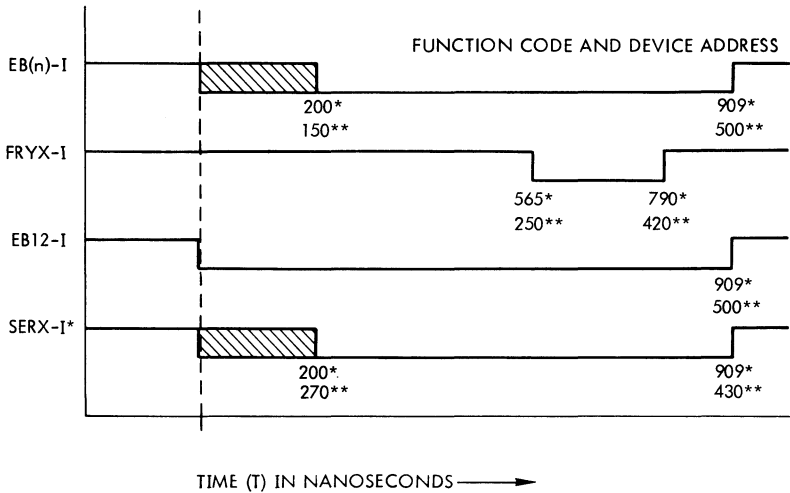
Logic levels: true = 0V dc,
false = +3V dc.

 = time when signal is settling.

* 620/f-100 timing

** 620/l-100 timing

Figure 7-5 External Control Timing



$T = 0$ is the start of the execute phase of the sense instruction.

Logic levels: true = 0V dc,
false = +3V dc.

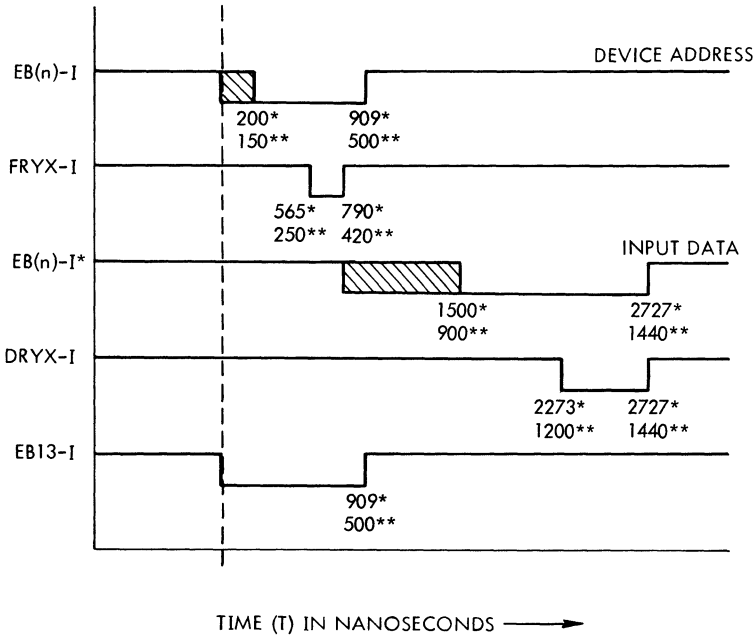
 = time when signal is settling.

* 620/f-100 timing

** 620/l-100 timing

*** For 620/f-100, SERX-I is normally on at T_{200} ; it must be on by T_{340} . For the 620/l-100, SERX-I is normally on at T_{270} ; it must be on by T_{410} .

Figure 7-6 Sense Response Timing



$T = 0$ is the start of the execute phase of the data transfer in instruction.

Logic levels: true = 0V dc,
false = +3V dc.

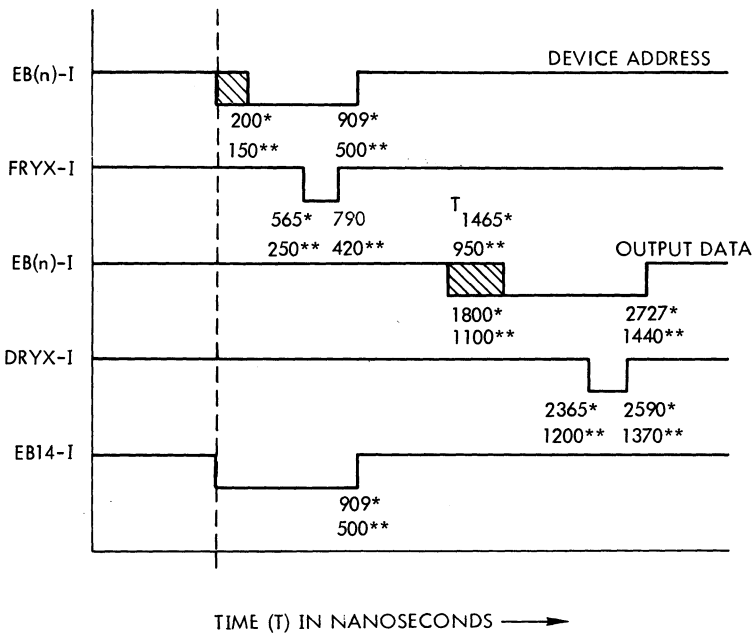
 = time when signal is settling.

* 620/f-100 timing

** 620/l-100 timing

*** For 620/f-100, E B(n)-I (input data) must be left off by T_{2727} . For the 620/l-100, E B(n)-I must be off by T_{1600} .

Figure 7-7 Data-Transfer-In Timing



T = 0 is the start of the execute phase of the data transfer out instruction.

Logic levels: true = 0V dc,
false = +3V dc.

 = time when signal is settling.

* 620/f-100 timing

** 620/l-100 timing

Figure 7-8 Data-Transfer-Out Timing

Interrupt-Initiated I/O

The Varian 620/f-100 I/O system includes an interrupt capability by which certain devices and I/O options, on a priority basis, can request the computer to execute an instruction (or a series of instructions) independent of the program in progress. During an interrupt, the computer is directed to a memory address specified by the interrupting device and executes the instruction at that address. Normally, the instruction at the interrupt address is a jump-and-mark instruction that results in the processing of an I/O service subroutine. The computer returns to the original program through an appropriate jump instruction at the conclusion of the interrupt subroutine.

Standard Varian 620/f-100 peripheral controllers normally are not capable of generating an interrupt because they cannot

provide the necessary memory address. The PIM supplies this addressing capability (section 8), thus implementing an **external interrupt system** within the computer interrupt system. A peripheral controller connected to a PIM directs an interrupt request to the PIM, which in turn sets IURX-I true and places the appropriate interrupt address on the I/O bus. The PIM then waits for the computer to acknowledge the request (IUAX-I true).

Up to eight interrupt levels can be serviced by one PIM. PIM priority logic establishes, on a hard-wired basis, the order in which interrupt requests are serviced. Normally the priority assignment is wired at the factory before equipment delivery. The user can, however, change this system to suit specific system requirements.

Standard Varian 620-100 interrupt addresses are:

620/f-100 Address	620/L-100 Address	Device and Function
020,21		MP halt error
032,033		MP I/O and overflow error
034,035		MP write and overflow error
036,037		MP jump and overflow error
040,041	040,041	Power failure
042,043	042,043	Power restart
044,045	044,045	RTC interval
046,047	046,047	RTC overflow
024,025	120,121	MP write error
026,027	124,125	MP jump error
022,023	130,131	MP I/O error
030,031	134,135	MP overflow error
100-117	100-117	PIM, first modules
120-177	140-177	PIM, remaining modules

Note: In the 620/L-100 if the MP is not included in the system, second and suc-

ceeding PIM interrupt addresses are 0120 through 0177.

Cycle-Stealing I/O

Direct-memory-access (DMA), cycle-stealing I/O operations are implemented by the addition of one or more (up to four) BICs (section 8). Cycle-stealing I/O combines the features of program-controlled and interrupt-initiated I/O. This mode of operation allows peripherals on the I/O bus to transfer data to or from memory while temporarily halting the processing of the stored program. This process is also referred to as "trapping".

Trap requests differ from interrupt requests in two ways:

- a. Interrupts direct the computer to the address of a subroutine, whereas the trapping requests require the computer to transfer data to or from memory. The data to be transferred is placed on the E bus after the memory address has been transmitted.
- b. The subroutine specified by an interrupt returns the computer to the main program, whereas trapping operations repeatedly halt the program while one word of data is transferred, then automatically returns control to the stored program. To preclude excessive preemption of stored program time, one program instruction is executed between successive traps. This method allows data to be transferred between memory and peripherals at rates up to 275,000 words per second.

Cycle-stealing traps do not disturb the contents of the operation registers (A, B, X, and P), thus freeing the CPU to perform other operations during data transfers.

Trapping operations can be initiated by the stored program or by a trap request from a peripheral controller under the control of a BIC. In either case, the BIC service subroutine establishes the initial and final addresses for the transfer, identifies the peripheral controller, and initializes both the selected controller and the BIC.

When the BIC senses that the peripheral controller is ready to transmit or accept data, it requests a trap, and, after receiving an acknowledgement from the computer, places the initial memory address on the E bus and increments the initial address buffer by one. During the transfer of the data word, the BIC is sensing the state of the peripheral controller. When more data are ready for transfer, the sequence is repeated until the initial address buffer contents equal the final address. The BIC then directs the computer back to the stored program.

Device Addresses

Standard device addresses assigned to options and peripherals used in Varian 620/L-100 I/O system operations are listed in table 7-2 grouped according to their function (i.e., class).

Table 7-2. Standard Device Addresses

Class Code	Addresses	Option or Peripheral
00-07	01-07	Teletype (620-06, -07, -08), or CRT device
010-017	010-013	Magnetic tape unit (620-30, -31)
	014	Fixed-head rotating memory (620-38, -42 through -49)
	015	Movable-head rotating memory (620-35)
	016,017	Movable-head rotating memory (620-37, -36)
020-027	020,021	First BIC (620-20)
	022,023	Second BIC
	024,025	Third BIC
	026,027	Fourth BIC
030-037	030	Card reader (620-25)
	031	Card punch (620-27)
	032	Digital plotter (620-72)
	033	Electrostatic plotter
	034	Second paper tape system
	035,036	Line printer (620-77, or 620-74)
	037	First paper tape system (620-53, -55, -55A, -51, -51A)
040-047	040-043	PIM (620-16)
	044	All PIM enable/disable
	045	MP (620-05)
	047	RTC (620-13)
050-047	050-053	Special applications, and Digital-to-analog converter (620-870 through -875)
	054-057	Analog system (620-85A, -850, -851)
060-067	060-067	Digital I/O controller (620-81), or Buffered I/O controller (620-80)
070-077	070-073	Data communications system (620-60, -61, -65, -66, -68)
	074-077	Relay I/O controller (620-83), or Special applications

SECTION 8 - I/O OPTIONS

To enhance I/O system operation, three I/O options are available with the Varian 620/f-100 computer:

- The Priority Interrupt Module (PIM)
- The Buffer Interlace Controller (BIC)
- The Priority Memory Access (PMA)

Interconnection details for the I/O options are given in section 11.

Priority Interrupt Module

The PIM (Model 620/f-116) provides for the orderly servicing of peripheral-initiated interrupts of a program in progress in the computer. It does so by:

- a. Establishing up to eight levels of interrupt priority for selected peripheral controllers.
- b. Storing interrupt requests originated by associated peripheral controllers and placing the requests on the I/O bus in the order of the established priority.

In effect, the PIM organizes a "priority-within-a-priority" system. Peripheral controllers that cannot normally initiate an interrupt because of their inability to generate memory addresses can do so when connected, via a line in the interrupt cable, to the PIM (which, as described in section 7, is included in the system interrupt priority chain). PIM-controlled priority assignments are prewired at the factory to user specifications.

Functional Description

Communication between the PIM and Varian 620/f-100 is similar to that of a peripheral controller, with the exception that the PIM can request a program interrupt.

The PIM automatically scans the system interrupt lines every 900 nanoseconds. When it detects an interrupt request from one of the connected peripheral controllers, the PIM relays the request to the computer. When the computer acknowledges the request, a predetermined interrupt address is placed on the I/O bus, and the computer is directed to execute the instruction at that address. Normally, the instruction at the interrupt address is a jump-and-mark operation that results in the processing of an I/O service subroutine; it can be, however, any instruction in the Varian 620/f-100 repertoire (section 16), except one from the I/O group.

If the PIM detects controller-initiated interrupt requests on more than one interrupt line, the highest priority request is accepted. Other requests are stored in the PIM's interrupt register and serviced in order of their priority.

The eight-bit PIM mask register can be used to disable any or all of the eight interrupt lines. This register is normally clear and is loaded only for disabling an interrupt. The mask register is not cleared automatically, however; it must be cleared and loaded by appropriate instructions in the program.

I/O OPTIONS

If the instruction at the interrupt address specified by the PIM is a jump and mark, PIM interrupts are inhibited until completion of the resultant service subroutine and subsequent program instructions reenable PIM operation.

PIM interrupts are also inhibited:

- a. During the execution of any two-word instruction
- b. During the execution of an I/O instruction
- c. After an I/O instruction until at least one non-I/O instruction has been executed
- d. During the execution of a shift instruction
- e. During a shift that is part of a multiplication or division operation
- f. Immediately following a shift during which a cycle-stealing I/O operation occurred
- g. During a manual execution of any instruction
- h. When the computer is in step mode (i.e., halted)
- i. During a manual or programmed halt
- j. Until the computer has fully entered the run mode
- k. During the processing of an execution instruction (whether conditional

or unconditional), including the execution of the instruction at the execution address

Since it is possible that portions of a program can contain only uninterruptable instructions, the insertion of at least one no-operation instruction ensures that imperative PIM interrupts (such as those initiated by the PF/R) can be recognized and serviced. In the case of an execution instruction, two such instructions are required.

Physical Description

The PIM is on one 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card, which is installed in an I/O slot of the memory or I/O expansion chassis.

Interface with the CPU is through the backplane wiring and I/O cable.

Peripheral controllers interface with the PIM through the backplane and/or I/O expansion cable. Controller priority assignments are prewired at the factory to user specifications.

PIM interconnection and interfacing is explained in greater detail in section 11. Refer also to the PIM maintenance manual (document number 98 A 9902 423).

Specifications

The physical, electrical, and functional specifications of the PIM are listed in table 8-1, and table 8-2 lists the PIM instructions.

Table 8-1. PIM Specifications

Parameter	Description
Organization	Contains line, synchronization, and mask registers; an interrupt address generator; priority and control logic; and line drivers and receivers
Control Capability	Establishes and implements eight levels of interrupt priority (user-assigned) for system peripherals
I/O Capability	Five external control and three transfer instructions
Standard Device Address	040 through 043
Interrupt Addresses	First PIM: 0100 through 0117 Second and succeeding PIMs: 0120 through 0177
System Priority Assignment	Determined by location in the 620-100 priority chain (user-selected)
Logic Level	
Internal	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc
I/O Cable	Negative logic: True: 0.0 to +0.45V dc False: +2.8 to +3.6V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Plugs into the 620-100 backplane; (additional PIMs interface via the I/O cable) connects to peripheral controllers via the backplane and/or I/O expansion cable
Input Power	+5V dc at 0.45 ampere
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 8-2. PIM Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 014x	10014x	Clear interrupt registers
EXC 024x	10024x	Enable PIM
EXC 034x	10034x	Clear interrupt registers and enable PIM
EXC 044x	10044x	Disable PIM
EXC 054x	10054x	Clear interrupt registers and disable PIM
EXC 0244	100244	Enable all PIMs
EXC 0444	100444	Transfer all PIMs
Transfer		
OME 04x	10304x	Transfer memory to mask register
OAR 04x	10314x	Transfer A register contents to mmask register
OBR 04x	10324x	Transfer B register contents to mask register
where x = device address (00 through 03).		

Buffer Interlace Controller

The BIC (Model 620-20) implements the transfer of blocks of data directly to and from computer memory and peripheral controllers. Cycle-stealing trap requests (section 7) inhibit the processing of a stored program for only the 1.7 microseconds (average) required to transfer one word of data directly between memory and a system peripheral. Operation register contents are not changed by the transfer, thus freeing the CPU to execute an instruction from the stored program between successive data word transfers.

The BIC monitors trap requests initiated by the stored program, or by magnetic-tape units, disc memories, card and paper-tape readers and punches, and analog-to-digital system controllers. Up to ten such devices can be connected to the BIC. The computer system can include up to four BICs.

BIC-monitored data transfers can occur at a maximum rate of 275,000 words per

second. Typically, the transfer rate *without* the BIC is 30,000 words per second.

Note that the BIC is included on the system priority chain; peripheral controllers connected to it have no priority of their own.

Functional Description

Data transfer control by the BIC is established by a user-coded service routine in the main program. The computer is instructed to sense the status of the BIC, and if it is not busy to initialize it to receive the initial and final memory addresses from/to which the block of data is to be transferred. The computer then senses the status of the selected peripheral, loads the initial and final address registers in the BIC, enables the BIC, and starts the peripheral controller. The BIC assumes control of the transfer, which is directly between the peripheral controller and computer memory via the E bus in the I/O cable.

Note that the data being transferred are not routed through the BIC. The BIC counts the number of data words that are transferred, and when the transfer is complete disconnects the peripheral controller and becomes not busy until again selected by the computer.

BIC-controlled data transfers can also be initiated by a trap request from a connected peripheral controller. The BIC senses when the controller is ready to transmit or accept data, then places a trap request on the E bus. When the computer acknowledges the request, the BIC places the initial memory address on the E bus and increments its initial address register by one until the initial and final address register contents are equal, signaling the completion of the transfer.

Termination of a data transfer can be requested by the peripheral controller in case of mechanical failure in the peripheral device, or when the number of data words to be transferred has not been previously determined and coded in the program.

The sequence controller circuit in the BIC generates the signals that coordinate address and data transmission between

the computer, BIC, and connected peripheral controllers.

Physical Description

The BIC is on one 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card, which is normally installed in a slot in the I/O section of the same chassis as the peripheral controllers to which it is connected.

Interface with the CPU is through the backplane wiring and I/O cable with control lines connecting the BIC and its associated peripheral controllers. If two or more BICs are installed in the same chassis, no control lines connect BICs. Control lines are extended via the I/O cable if the BIC and a connected peripheral controller are in separate chassis.

BIC interconnection and interfacing is explained in greater detail in section 11. Refer also to the BIC maintenance manual (document number 98 A 9902 113).

Specifications

The physical, electrical, and functional specifications of the BIC are listed in table 8-3, and table 8-4 lists the BIC instructions.

Table 8-3. BIC Specifications

Parameter	Description
Organization	Contains an initial address register, a final address register, sequence controller, and drivers and receivers
Control Capability	Implements trapping operations for up to ten peripheral controllers
I/O Capability	Two external control, eleven transfer, and two program sense instructions

Table 8-3. BIC Specifications (continued)

Parameter	Description
Transfer Rate	Synchronized with peripheral: 275,000 words per second (maximum)
I/O Signal Limits	Rise/fall: 10 nanoseconds (minimum), 100 nanoseconds (maximum)
System Priority	Determined by location in the 620-100 priority chain (user-selected)
Standard Device Address	First BIC: 020-021 Second BIC: 022-023 Third BIC: 024-025 Fourth BIC: 026-027
Logic Level	
Internal	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc
I/O Cable	Negative logic: True: 0.0 to +0.5V dc False: +2.8 to +3.6V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Plugs into the 620-100 backplane, with control lines between the BIC and associated peripheral controllers; controllers in a separate chassis connect to the BIC via control lines in the I/O cable
Input Power	+5V dc at 0.6 ampere
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 8-4. BIC Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0020	100020	Enable BIC
EXC 0021	100021	Initialize BIC
Transfer		
OAR 0120	103120	Transfer A register contents to initial address register
OBR 0220	103220	Transfer B register contents to initial address register
OME 0020	103020	Transfer memory contents to initial address register
OAR 0121	103121	Transfer A register contents to final address register
OBR 0221	103221	Transfer B register contents to final address register
OME 0021	103021	Transfer memory contents to final address register
INA 0120	102120	Transfer initial register contents to the A register
INB 0220	102220	Transfer initial register contents to the B register
IME 0020	102020	Transfer initial register contents to memory
CIA 0520	102520	Clear A register and transfer initial register contents
CIB 0620	102620	Clear B register and transfer initial register contents
Program Sense		
SEN 0020	101020	Sense BIC not busy
SEN 0021	101021	Sense abnormal device stoppage

Priority Memory Access (PMA)

The Model 620/f-112 Priority Memory Access (PMA) is a mainframe option for Varian 620/f-100 computer systems. The PMA option interfaces with PMA controllers to provide logic functions to interface four data transfer channels with the Varian 620/f-100 memory in a hardware-fixed priority. All signals are interlocked to

free the PMA option/PMA controller interface from circuit and cable speed dependence, and to provide the capability to handle a variety of circuits and data rates.

Specifications for the PMA option are given in table 8-5.

The PMA option can interface with up to eight PMA controllers distributed in any

Table 8-5. PMA Specifications

Parameter	Description
Organization	Contains control logic, priority logic, data transfer logic, data drivers, and input data receivers and address receivers
Priority Assignment	Exceeded only by the PF/R
Number of Channels	Four with hardware-assigned priority
Maximum Number of Controllers per PMA	Eight distributed in any manner among four channels (only one active controller per channel at a time)
Maximum Cable Length	20 feet
Typical Maximum Latency	1,085 nanoseconds
Typical Minimum Latency	378 nanoseconds
Typical Maximum Transfer Rate	1.33 MHz
Control Signals	All signals are controlled by interlocked responses allowing true asynchronous operation
Logic Levels	Positive logic
PMA Bus	True: + 2.2 to + 3.2V dc False: - 0.5 to + 1.4V dc
	Negative logic
	True: - 0.5 to + 1.4V dc False: + 2.2 to + 3.2V dc
Internal	Positive logic
	True: + 2.2 to + 3.2V dc False: - 0.5 to + 1.4V dc
Size	One 3-by-15-inch (7.7 x 38.1 cm) wired-socket card
Interconnection	Plugs into the 620/f-100 CPU tray which connects to the PMA bus connector J51 at the rear of the 620/f-100 mainframe

Table 8-5. PMA Specifications (continued)

Parameter	Description
Power Consumption	+5V dc at 2.0 amperes -5V dc at 300 mA +3.2V dc at 2.2 amperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

manner among the four channels although only one PMA controller per channel can be active at a time. Each PMA controller interfaces with the PMA option board that plugs into the Varian 620/f-100 mainframe CPU tray and with the Varian 620/f-100 I/O bus for program control (figure 8-1).

PMA controllers are either designed by the user or supplied by Varian. A Varian-supplied block-transfer controller (PMA BTC) is described at the end of this section.

The four PMA option channels share the 15-bit address bus, the 16-bit bidirectional bus, the READ line, the SYRT-P line, the PRMS line, and the GO line. Each channel has its own request line (REQx) and its own acknowledge line (ASKx). Access to the PMA option control and bus signals is through J51 at the rear of the Varian 620/f-100 mainframe.

The Control II Card in slot 10 of the mainframe CPU tray contains the memory access control logic which decides whether the PMA, DMA, or CPU gains access to memory. PMA has the highest priority, as shown in figure 8-2.

PMA Functional Description

The PMA is functionally divided into six circuits: control, priority, data transfer,

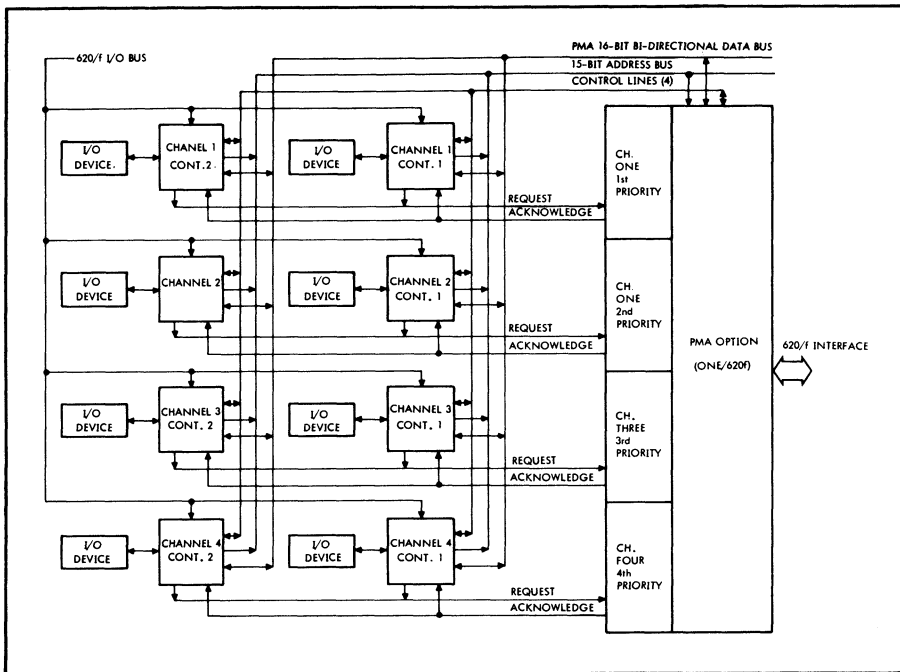
data drivers, input data receivers, and address receivers (figure 8-3).

The control logic provides enable terms to the other functional blocks, priority check and full interlock control of the four channels, and control signal interface with the Varian 620/f-100 memory access logic.

The priority logic assigns priority to the four PMA channels on a hardware basis. It controls the channel acknowledge lines and the PMA request (PMAR -) line to the Varian 620/f-100 memory access logic.

The four channel request (REQx) signals are gated directly to the memory access logic before priority is established to reserve the next memory cycle for the PMA. The priority logic constantly monitors the four channels until a request is received. The logic then establishes priority (channel 1 has the highest priority and channel 4, the lowest) and enables the proper acknowledge line. Upon receipt of the acknowledge signal, the PMA controller can disable the request or, if consecutive memory cycles are required, maintain the enabled condition.

The priority logic continues to monitor priority, thus, channel priority is reestablished between each PMA memory cycle upon receipt of memory Data Read (MDR). If a higher priority channel remains enabled, all lower priority channels are locked out.



VT12-0378

Figure 8-1. PMA/Controller Interface

The data transfer logic controls the flow of data through the PMA option and synchronizes priority determination with memory cycle completion.

Sixteen data drivers (PD00+ through PD15+) provide gating and interface between the memory output data bus and the PMA 16-bit bidirectional data bus.

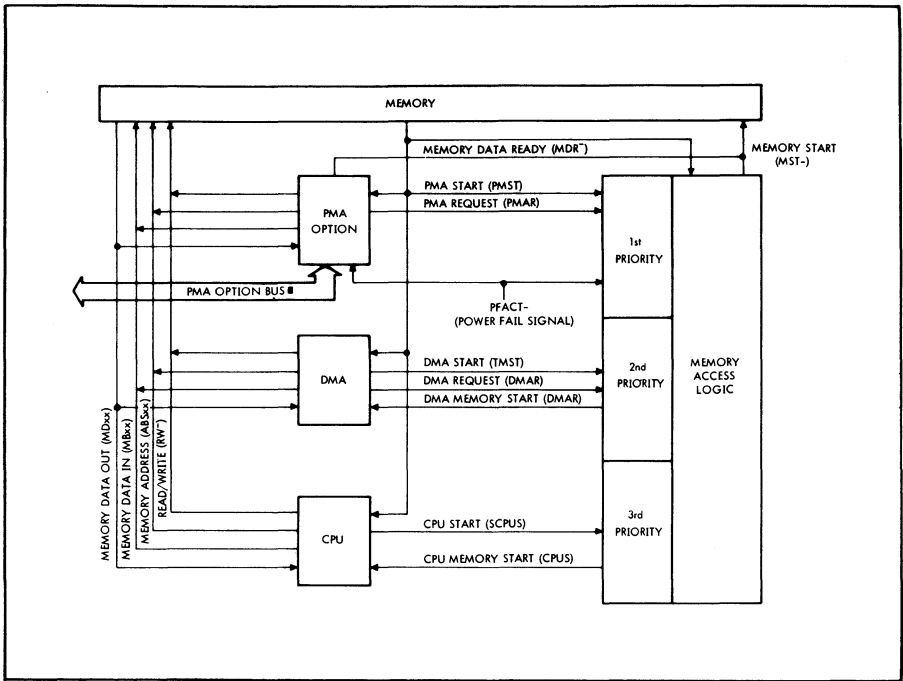
Sixteen input data receivers (MB00 – through MB15 –) provide gating and interface between the memory input data bus (MBxx) and the PMA 16-bit bidirectional data bus (PDxx).

Fifteen address receivers (ABS00 – through ABS14 –) provide gating and interface between the memory address bus (ABSxx –) and the PMA 15-bit address bus (PAXx).

PMA Optional/Controller Interface

PMA controllers can be installed in any of the six I/O card slots of an I/O expansion chassis.

PMA interconnection consists of routing signals from the PMA circuit board in card slot 2 of the CPU mainframe to the I/O backplane in the expansion chassis. The



VT12-0379

Figure 8-2. PMA, DMA, Control Line

expansion chassis backplane is wired specifically for PMA use.

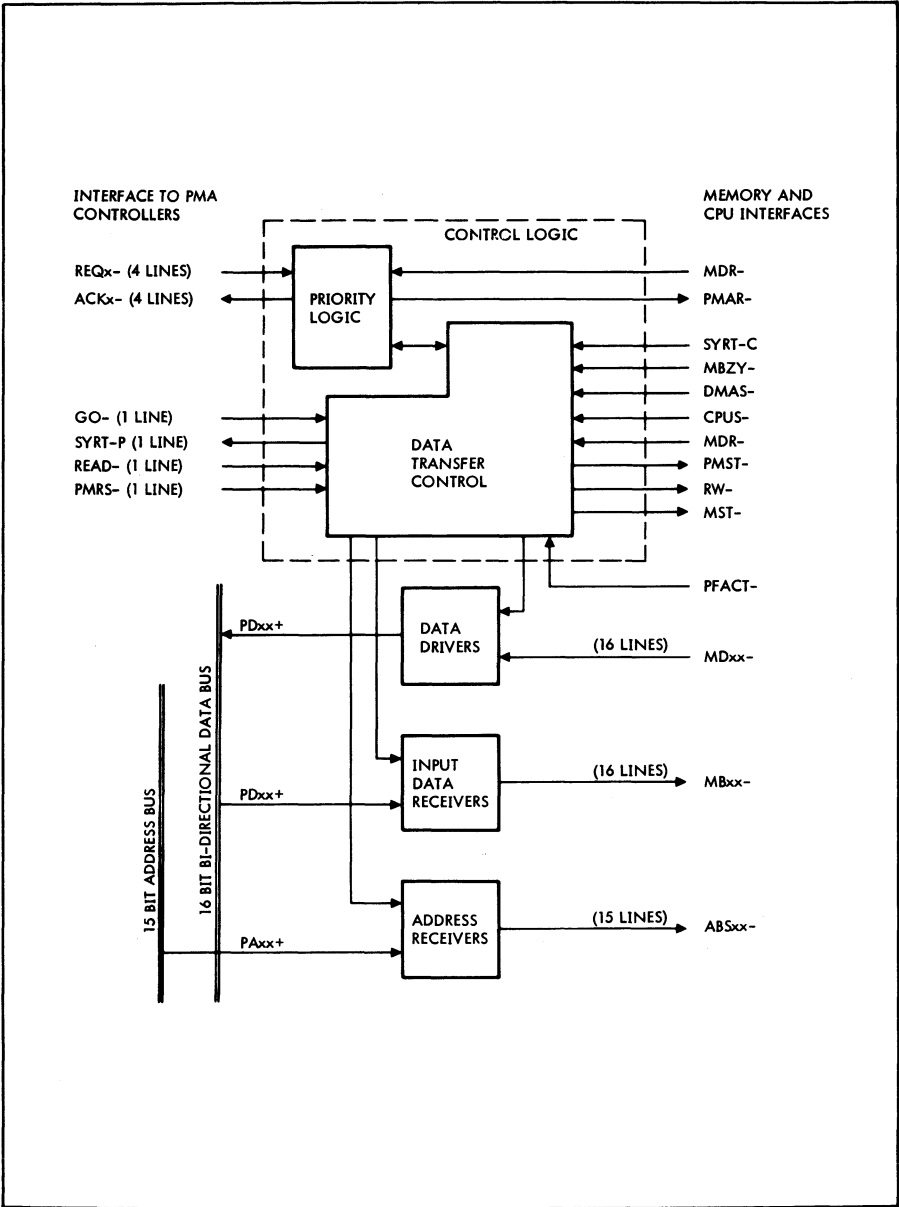
The PMA cable connects 50-pin card-edge connector J51 on the mainframe to circuit card connectors on the I/O backplane of the expansion chassis. The PMA lines on the I/O backplane must be terminated with a PMA termination shoe. The cable can be up to 20 feet long, if required.

The four PMA channels share the 15-bit address bus, the 16-bit bidirectional bus, the READ - line, the SYRT-P line, the PRMS - line, and the GO - line. Each channel has its own request (REQx) line and its own acknowledge (ACKx -) line.

PMA Option/Controller Interface Circuits

The PMA option/controller driver comprises two high-speed open-collector two-input NAND gates in parallel (SN74H01N). The parallel drivers are contained in the same integrated circuit package (part number 49A0042-000). User-designed controllers should also follow this practice to ensure equal current sharing by the parallel drivers when sinking current to the ground voltage level.

The PMA option/controller receiver is a dual differential line receiver (SN75107N or equivalent; part number 49A0114-000).



VTII-1017

Figure 8-3. PMA Functional Block Diagram

Refer to Figures 8-4, 8-5, and 8-6 for typical circuit descriptions.

The PMA cable is constructed with twisted-pair 24- or 226-gage wire and has a maximum length of 20 feet. A maximum of two controller chassis are packaged at a minimum distance apart to minimize discontinuity effects in the PMA cable. All lines in the cable are terminated at both ends. Termination resistors are mounted on the PMA option board, and termination for the other end of the cable is provided on the last controller or by a termination shoe located at the end of the cable.

PMA Block Transfer Controller

The PMA Block Transfer Controller (BTC) interfaces the 620/f-100 PMA option to a peripheral controller requiring fast access, high-speed transfer of data in or out of processor core memory. The BTC provides memory address control when the transferred data are organized into "blocks" of 16-bit words.

The BTC buffers one word of data in and out of the PMA option so that the peripheral controller timing requirements can be relaxed with respect to the fast PMA bus.

Peripheral controllers used with the BTC must be designed for PMA use since the PMA bus is separate from the E-bus shared by I/O and DMA.

As can be seen in figure 8-7, the BTC interfaces to three points: the PMA option, the I/O bus, and the PMA controller.

Under program control, the BTC is loaded with the initial memory address and the final memory address of the block transfer. The BTC is then connected to the peripheral controller to enable it to connect to the PMA port through the BTC.

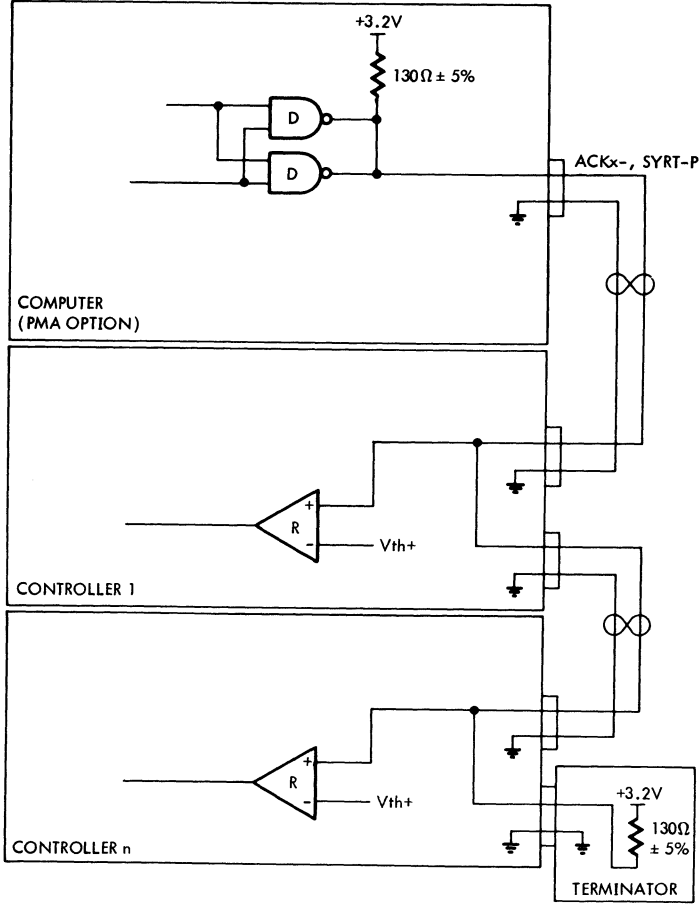
A PMA transfer cycle is initiated by the peripheral controller issuing a PMA Request which the BTC passes on to the PMA option board in the CPU. When the PMA Option has captured the memory from the DMA and CPU, PMA Acknowledge is returned to the BTC and peripheral controllers. The memory then sits idle waiting for PMA GO from the peripheral controller.

When PMA GO is received at the PMA Option, the memory address and read/write line from the BTC are gated into the memory to start the cycle. Typically, the peripheral controller is ready to transfer data when it raises a PMA Request. This means it can wrap-around PMA Acknowledge as PMA GO with a minimum of delay.

As the PMA cycle occurs, the BTC buffers the transferred data, and increments its memory address counter for the next cycle. If the peripheral controller sustains its request, it can retain access to memory through the PMA port and receive consecutive memory cycles.

When the peripheral controller has received or sent enough words for its purpose, it removes its request to stop the PMA process. The BTC stops incrementing its memory address and the CPU program resumes at the point where the peripheral request was acknowledged. The peripheral controller remains connected to the BTC. When the accumulation of PMA word transfers or one long PMA transfer reaches the block length initially loaded into the BTC, the BTC unconditionally disconnects from the peripheral controller and PMA port and generates a BTC finished term which can be used to set a PIM interrupt.

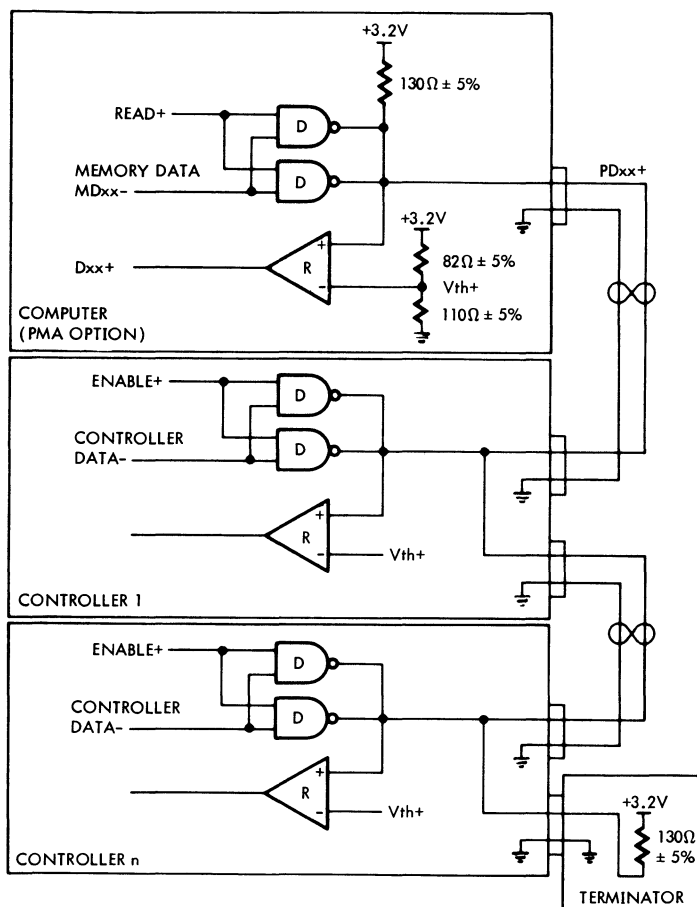
The BTC cannot be reconnected until it is reloaded with the parameters of the next transfer.



NOTES: D = SN74H01 GATE OR EQUIVALENT
R = SN75107 DIFFERENTIAL RECEIVER OR EQUIVALENT
 $V_{th+} = +1.8V \pm 5\%$

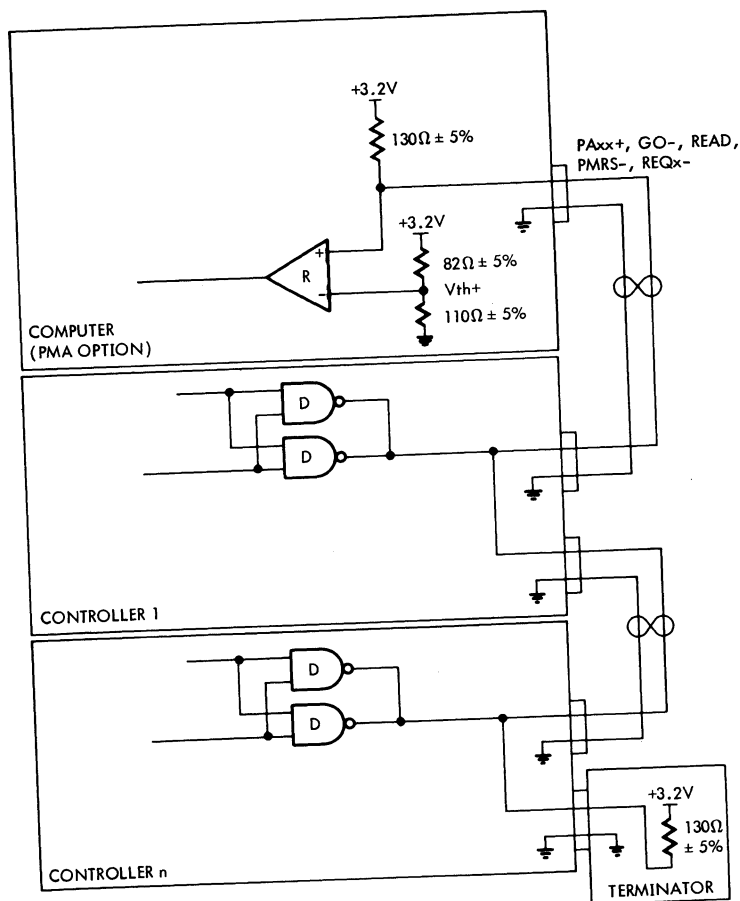
VT11-1018

Figure 8-4. Typical PMA Control Line Out



NOTES: D = SN74H01 GATE OR EQUIVALENT
 R = SN75107 DIFFERENTIAL RECEIVER OR EQUIVALENT
 $V_{th+} = +1.8V \pm 5\%$

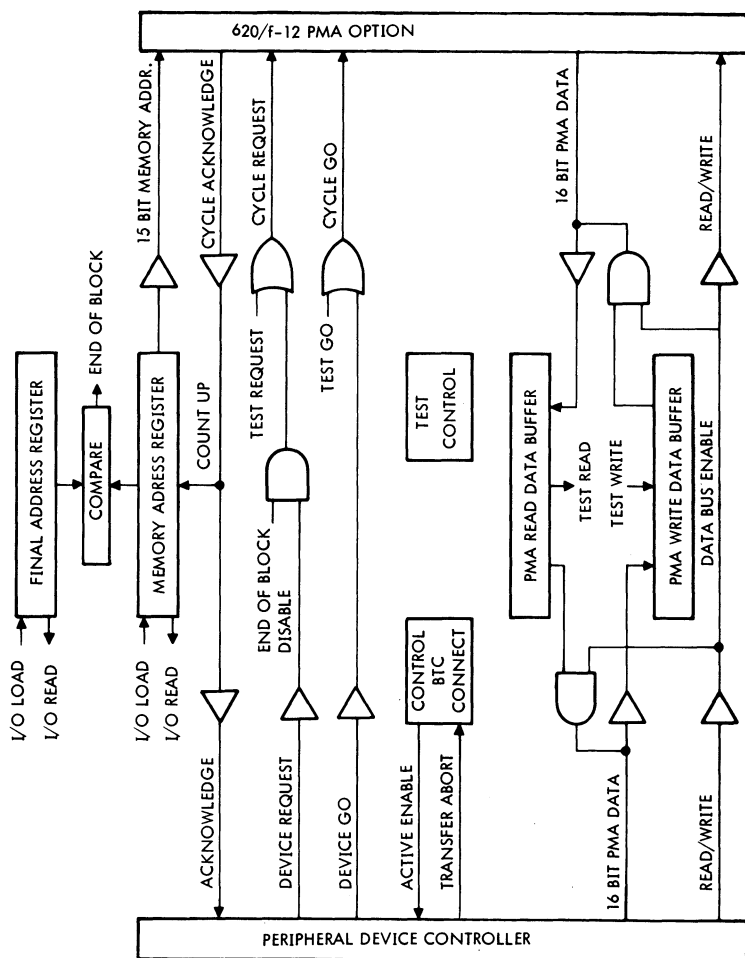
Figure 8-5. Typical PMA 16-Bit Bidirectional Bus Line



NOTES: D = SN74H01 GATE OR EQUIVALENT
 R = SN75107 DIFFERENTIAL RECEIVER OR EQUIVALENT
 $V_{th+} = +1.8V \pm 5\%$

VT11-1020

Figure 8-6. Typical PMA Control Line In or Address Line



VT11-1287

Figure 8-7. BTC Block Diagram

I/O OPTIONS

The peripheral controller may abort a transfer by initiating a disconnect to the BTC. The peripheral controller must inform the program of the reason for the abort so that the transfer may be redone in an orderly fashion.

Note: Any abort action by peripheral controller must not occur during a PMA/BTC memory cycle.

BTC Programming

PMA BTC Programming is identical to that of the DMA BIC except that the data

transfer occurs much faster. Additional instructions are also provided, mainly to serve as diagnostic aids in testing the BTC, PMA bus, and PMA options.

The PMA BTC instructions are given in table 8-6.

Like the DMA BIC, the PMA BTC uses a pair of device addresses - $2x$ and $2(x+1)$, from the group 20 through 27. The possible even/odd address pairs are 20/21, 22/23, 24/25, and 26/27.

Table 8-6. PMA BTC Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 002x	100020	Active enable
EXC 002 (x + 1)	100021	Initialize
EXC 012x	100120	PMA port reset
EXC 012 (x + 1)	100121	Test data transfer
EXC 022x	100220	Test cycle write
EXC 022 (x + 1)	100221	Test cycle read
Program Sense		
SEN 002x	101020	Sense BTC not busy
SEN 002 (x + 1)	101021	Sense abnormal device stop
SEN 012x	101120	Test sense end of block
Transfer		
OAR 002x	103120	Load BTC initial address register*
OBR 002x	103220	Load BTC initial address register*
OME 002x	103020	Load BTC initial address register*
OAR 002 (x + 1)	103121	Load BTC final address register
OBR 002 (x + 1)	103221	Load BTC final address register
OME 002 (x + 1)	103021	Load BTC final address register
INA 002x	102120	Read initial address register**

Table 8-6. PMA BTC Instructions (continued)

Mnemonic	Octal Code	Description
INB 062x	102220	Read initial address register**
IME 002x	102020	Read initial address register**
CIA 052x	102520	Read initial address register**
CIB 062x	102620	Read initial address register**
INA 012 (x + 1)	102121	Read final address register
INB 022 (x + 1)	102221	Read final address register
IME 002 (x + 1)	102021	Read final address register
CIA 052 (x + 1)	102521	Read final address register
CIB 062 (x + 1)	102621	Read final address register

* When immediately preceded by an EXC 012 (x + 1) instruction (Test Data Transfer), the output instruction loads the PMA Write Data Buffer from the E-Bus. The actual transfer clears the Test EXC logic. The buffer can then be transferred into memory by PMA with a Test Cycle Write instruction.

** When immediately preceded by an EXC 012 (x + 1) instruction (Test Data Transfer), the input instruction will read the contents of the PMA Read Data Buffer back over the E bus. The actual transfer clears the Test EXC logic. The buffer can be read in this fashion after a Test Cycle Read instruction has loaded the buffer through PMA.

SECTION 9 – PERIPHERALS AND I/O INTERFACES

A complete line of peripherals and I/O interfaces is available for use with the Varian 620-100 computers. This section presents descriptions of a sampling of the peripheral device/controller combinations offered.

- Teletypes
- Paper Tape System
- Magnetic Tape Equipment
- Disc Memories
- Drum Memories
- Line Printers
- Punched Card Equipment
- Oscilloscope Display Units
- Digital Plotters
- Analog I/O Systems
- Data Communications Equipment
- Buffered I/O Controllers
- Relay I/O Controllers
- Digital I/O Controllers
- Disc Memory System
- Universal Controller

The circuits of the Varian 620-100 series peripheral controllers and I/O interfaces are contained on 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit cards that can be installed in any peripheral controller slots in the expansion chassis.

This section briefly discusses peripheral device and controller capabilities. Refer to the applicable Varian Data Machines technical manual for details of peripheral specifications, installation, operation, and maintenance.

Teletypes

The peripheral controller for the first Teletype in the Varian 620-100 system is assigned a slot in the mainframe. It interfaces with the CPU through the memory chassis back-plane and memory address cable; refer to section 12 and 34 for interconnection details.

If more than one Teletype is required, peripheral controllers (Models 620-06, 07, 08) can be installed in any I/O slot in the expansion chassis. The additional Teletypes also require the Teletype buffer board (Model 620-09). As many as eight Teletypes can be included in one 620-100 system. (All teletype controllers in the 620/L-100 can be installed in any I/O slot in the main frame or expansion chassis).

The factory-modified Teletype unit controlled by the Teletype controller can be a Teletype Model 33 ASR, 35 ASR, or 35 KSR. The ASR models include paper-tape reader and punch capabilities; the KSR model uses only keyboard-entered instructions and data.

Specifications of the Teletype controller are given in table 9-1, and table 9-2 lists the Teletype instructions. Refer also to the Teletype controller manuals (document numbers 98 A 9902 162 and 98 A 9908 160).

Table 9-1. Teletype Controller Specifications

Parameter	Description
Organization	Contains input and output registers, timing control circuitry for simultaneous two-way communication, and CPU/Teletype interface logic
Control Capability	One factory-modified Teletype Model 33 ASR, 35 ASR, or 35 KSR, including cable
I/O Capability	Three external control, eight transfer, and two program sense instructions
Operation Modes	Input: from keyboard or paper-tape reader Output: to Teletype printer or paper-tape punch
Interrupt Capability	Ready to write and ready to read interrupt available to PIM
Standard Device Address	01 through 07
Memory Access Control	By CPU indirectly (via Teletype buffer board), or by BIC (second and subsequent controllers).
Logic Levels	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc
Size	First card is a 3.1 by 14.9 (37.3 x 7.8 cm). Second and subsequent cards are 7-3/4 by 12-inch (19.7 x 30.3 cm)*
Interconnection	Interfaces with the CPU, I/O bus or BIC, and Teletype unit via memory chassis backplane connector
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

*All 620/L-100 cards are 7-3/4 by 12 inch (19.7 x 30. cm)

Table 9-2. Teletype Controller Instructions

Mnemonic	Octal Code	Description	
External Control			
EXC 0101	100101	Connect input register to BIC	
EXC 0201	100201	Connect output register to BIC	
EXC 0401	100401	Initialize Teletype Controller	
Transfer			
OAR 0101	103101	Transfer A register contents to input register	
OBR 0201	103201	Transfer B register contents to input register	
OME 0001	103001	Transfer memory contents to input register	
INA 0101	102101	Transfer output register contents to A register	
INB 0201	102201	Transfer output register contents to B register	
IME 0001	102001	Transfer output register contents to memory	
CIA 0501	102501	Transfer output register contents to cleared A register	
CIB 0601	102601	Transfer output register contents to cleared B register	
Program Sense			
SEN 0101	101101	Ready to write	
SEN 0201	101201	Ready to read	
Teletype Commands			
Function	Symbol	Octal Code	Type As:
Enable printer	SOM	201	CONTROL and A
Suppress printer	EOT	204	CONTROL and D
Reader on	XON	221	CONTROL and Q
Punch on	TAPE	222	CONTROL and R
Reader off	XOFF	223	CONTROL and S
Punch off	TAPE OFF	224	CONTROL and T

High-Speed Paper Tape Equipment

The high-speed paper tape system (Model 620-55A) comprises a controller, perforator, and reader. A paper tape spooler is also available for use with the reader.

The controller card contains a data register that buffers the data words being transferred, a decoder section that interprets instructions received from the computer, a timing and control section that synchronizes operation of the peripheral equipment with the computer, and necessary interface hardware.

The controller can transfer data from the reader to the computer. It can also transfer data to the perforator from the computer. It can be used to reproduce paper tapes. The controller can transfer data

into the computer in a continuous read mode, which places the reader in continuous slew until an instruction to stop is received, or it can operate in a step read mode, requiring a new instruction from the computer for each transmitted data word.

Computer control of the paper tape system is accomplished through the I/O bus. The controller can also be operated under the direction of the BIC.

Each controller is capable of operating one perforator and one reader on a time-shared basis.

Specifications of the paper tape system controller are given in table 9-3, and table 9-4 lists the paper tape system instructions. Refer also to the paper tape controller manual (document number 98 A 9902 152).

Table 9-3. High-Speed Paper Tape Controller Specifications

Parameter	Description
Organization	Contains a timing and control section, instruction decoder, and an eight-bit data buffer register
Control Capability	One reader and one perforator, operated on a time-shared basis
I/O Capability	Five external control, eight transfer, and one program sense instructions
Operational Modes	Continuous read: 300 characters per second Step read: 1 to 300 characters per second Punch: 1 to 75 characters per second
Standard Device Address	037

continued

Logic Levels	Positive logic: True: +2.5 to +5.0V dc False: 0.0 to +0.5V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Interfaces with the computer and BIC via a 122-terminal connector; interfaces with punch and reader via two 44-terminal connectors
Input Power	+5V dc at 540 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-4. High-Speed Paper Tape Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0037	100037	Connect punch to BIC
EXC 0437	100437	Stop reader and initialize controller
EXC 0537	100537	Start reader
EXC 0637	100637	Enable punch buffer
EXC 0737	100737	Read one character
Transfer		
OAR 0137	103137	Transfer A register contents to data buffer register
OBR 0237	103237	Transfer B register contents to data buffer register
OME 0037	103037	Transfer memory contents to data buffer register
INA 0137	102137	Transfer data buffer contents to A register
INB 0237	102237	Transfer data buffer contents to B register
IME 0037	102037	Transfer data buffer contents to memory
CIA 0537	102537	Transfer data buffer contents to cleared A register
CIB 0637	102637	Transfer data buffer contents to cleared B register
Program Sense		
SEN 0537	101537	Sense buffer ready

Magnetic Tape Equipment

Seven-Track Magnetic Tape Equipment

The seven-track magnetic tape system consists of up to four magnetic tape transports and a controller (Model 620-31A,-31B,-31C). The system can read and record a magnetic tape that is IBM-2400-compatible for seven-track systems.

The magnetic tape controller provides a buffered interface between the I/O bus and a seven-track magnetic tape transport. The controller accommodates up to four transports of the Peripheral Equipment Corporation 6000 series. However, only one transport is selected for use at any one time.

The controller comprises two circuit cards and contains all read/write data buffer registers and timing and control logic required to control one tape transport.

Computer control of the magnetic tape system is accomplished through the I/O bus. The controller can also be operated under the direction of the BIC.

When more than one tape transport is used with the controller, the transports are connected to the controller in party-line configuration. However, only one transport can be operated at a time. Transport selection is under the control of the stored program.

The controller specifications are given in table 9-5, and table 9-6 lists the seven-track magnetic tape system instructions. Refer also to the seven-track magnetic tape controller manual (document number 98 A 9902 140).

Nine-Track Magnetic Tape Equipment

The nine-track magnetic tape system consists of up to four magnetic tape transports and a controller (Model 620-30). The system can read and record a magnetic tape that is IBM-2400-compatible for nine-track systems.

The magnetic tape controller provides a buffered interface between the I/O bus and a nine-track magnetic tape transport. The controller accommodates up to four transports, but only one transport is in use at any one time.

The controller comprises two circuit cards and contains all read/write data buffer registers and timing and control logic required to control one tape transport.

Computer control of the magnetic tape system is accomplished through the I/O bus. The controller can also be operated under the direction of the BIC.

When more than one tape transport is used with the controller, the transports are connected to the controller in party-line configuration. However, only one transport can be operated at a time. Transport selection is under the control of the stored program.

The controller specifications are given in table 9-7, and table 9-8 lists the nine-track magnetic tape system instructions. Refer also to the nine-track magnetic tape controller manual (document number 98 A 9902 121).

Table 9-5. Seven-Track Magnetic Tape Controller Specifications

Parameter	Description
Organization	Contains instruction decoder and storage logic, sense logic, read/write data controller, write/read motion controller, read/write data storage and error-checking logic, and timing and control logic
Control Capability	Four tape transports, any one of which can be selected for connection; system reset automatically selects transport 1
I/O Capability	Eight external control, eight transfer, eight program sense, and four transport selection instructions
Data Word	Provides buffering for two 16-bit words, each containing up to three bytes
Error-Checking	Longitudinal redundancy check (LRC) and cyclic redundancy check (CRC) characters regenerated during reading; correction not provided
Standard Device Address	010 through 013
Logic Levels	Positive logic: True: +2.5 to +5.0V dc False: 0.0 to +0.5V dc
Size	Two 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket cards
Interconnection	Each card interfaces with the computer and BIC via a 122-terminal connector; two 44-terminal connectors interface each card with the tape transport
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-6. Seven-Track Magnetic Tape Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0010	100010	Read one binary record
EXC 0110	100110	Read one BCD record
EXC 0210	100210	Write one binary record
EXC 0310	100310	Write one BCD record
EXC 0410	100410	Write a file mark
EXC 0510	100510	Skip forward one record
EXC 0610	100610	Backspace one record
EXC 0710	100710	Rewind
Transfer		
OAR 0110	103110	Transfer A register contents to buffer register
OBR 0210	103210	Transfer B register contents to buffer register
OME 0010	103010	Transfer memory contents to buffer register
INA 0110	102110	Transfer buffer register contents to A register
INB 0210	102210	Transfer buffer register contents to B register
IME 0010	102010	Transfer buffer register contents to memory
CIA 0510	102510	Transfer buffer register contents to cleared A register
CIB 0610	102610	Transfer buffer register contents to cleared B register
Program Sense		
SEN 0010	101010	Sense parity error
SEN 0110	101110	Sense buffer ready
SEN 0210	101210	Sense magnetic tape unit ready
SEN 0310	101310	Sense file mark
SEN 0410	101410	Sense odd-length record/high density (optional)
SEN 0510	101510	Sense end of tape
SEN 0610	101610	Sense beginning of tape
SEN 0710	101710	Sense rewinding

continued

Transport Selection

EXCB 0110	104110	Select tape drive 1
EXCB 0210	104210	Select tape drive 2
EXCB 0310	104310	Select tape drive 3
EXCB 0410	104410	Select tape drive 4

Table 9-7. Nine-Track Magnetic Tape Controller Specifications

Parameter	Description
Organization	Contains instruction decoder and storage logic, sense logic, read/write data controller, read/write motion controller, read/write data storage and error-checking logic, and timing and control logic
Control Capability	Four tape transports, any one of which can be selected for connection; system reset automatically selects transport 1
I/O Capability	Eight external control, eight transfer, eight program sense, and four transport selection instructions
Data Word	Provides buffering for two 16-bit words, each containing two 8-bit bytes
Error-Checking	Longitudinal redundancy check (LRC) and cyclic redundancy check (CRC) characters regenerated during reading; correction not provided
Standard Device Address	010 through 013
Logic Levels	Positive logic: True: +2.5 to +5.0V dc False: 0.0 to +0.5V dc
Size	Two 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket cards
Interconnection	Each card interfaces with the computer and BIC via a 122-terminal connector; two 44-terminal connectors interface each card with the tape transport
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-8. Nine-Track Magnetic Tape Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0010	100010	Read one binary record
EXC 0110	100110	Read one BCD record
EXC 0210	100210	Write one binary record
EXC 0310	100310	Write one BCD record
EXC 0410	100410	Write a file mark
EXC 0510	100510	Skip forward one record
EXC 0610	100610	Backspace one record
EXC 0710	100710	Rewind
Transfer		
OAR 0110	103110	Transfer A register contents to buffer register
OBR 0210	103210	Transfer B register contents to buffer register
OME 0010	103010	Transfer memory contents to buffer register
INA 0110	102110	Transfer buffer register contents to A register
INB 0210	102210	Transfer buffer register contents to B register
IME 0010	102010	Transfer buffer register contents to memory
CIA 0510	102510	Transfer buffer register contents to cleared A register
CIB 0610	102610	Transfer buffer register contents to cleared B register
Program Sense		
SEN 0010	101010	Sense parity error
SEN 0110	101110	Sense buffer ready
SEN 0210	101210	Sense magnetic tape unit ready
SEN 0310	101310	Sense file mark
SEN 0410	101410	Sense odd-length record/high density (optional)
SEN 0510	101510	Sense end of tape
SEN 0610	101610	Sense beginning of tape
SEN 0710	101710	Sense rewinding
Transport Selection		
EXCB 0110	104110	Select tape drive 1
EXCB 0210	104210	Select tape drive 2
EXCB 0310	104310	Select tape drive 3
EXCB 0410	104410	Select tape drive 4

Disc Memories

Disc memory systems consist of a rotating memory (Models 620-38A, -38B, -38C, and 620-37) and a controller. The 620-38 models are of the fixed-head-per-track type; the 620-37 model is a high-capacity, moving-head disc unit. Both types can be rack-mounted.

Disc memories offer bulk storage for data and library software routines. Data can be stored and retrieved at maximum data transfer rates of up to 80K words per second.

The hardware components comprising a

complete operating system for both disc memories are:

- a. Rack-mounted disc memory unit containing a built-in operating dc power supply
- b. Interconnecting system cable between computer chassis and disc
- c. A disc controller to supply a plug-in interface

Specifications for the 620-38 discs are given in table 9-9, and for the 620-37 model, in table 9-10. Disc controller instructions are listed in table 9-11.

Table 9-9. Model 620-38 Disc Memory Specifications

Parameter	Description
	Disc Memory Unit
Organization	One fixed head per track; 16, 32, or 64 tracks with 1,920 words per track
Transfer Rate	72,000 words per second (60 Hz), 60,000 words per second (50 Hz)
Rotational Speed	1,775 rpm (60 Hz); 1,480 rpm (50 Hz)
Recoverable Access Time	17 milliseconds (60 Hz), 20 milliseconds (50 Hz)
Write-to-Read Recovery Time	Less than 10 microseconds
Dimensions	10-1/2 inches (26.9 cm) high, 19 inches (48.7 cm) wide, and 19 inches (48.7 cm) deep (mounts in standard RETMA rack)
Weight	Approximately 50 pounds (22.7 kg)

continued

PERIPHERALS AND I/O INTERFACES

Primary Power	120 $\pm$ 12V ac (60 Hz), single phase, 8 amperes starting current, and 3.5 amperes running current; 220V ac (50 Hz), 4 amperes starting current, and 2.25 amperes running current
Vibration	10g in 20 to 100 Hz range
Operational Environment	5 to 45 degrees C; 20 to 90 percent relative humidity without condensation
Control Capability	Disc Controller One Model 620-38A,-38B,-38C disc memory unit
I/O Capability	Two external control, three transfer, and seven program sense instructions
Standard Device Address	014
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card
Interconnection	Connects to disc unit via a 10-foot (3.07m) interface cable; interfaces with the computer and BIC via a 122-terminal connector
Input Power	+5V dc at 10 watts (nominal) supplied by the computer power supply
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-10. Model 620-37 Disc Memory Specifications

Parameter	Description
Organization	Disc Memory Unit Moving-head, single-disc, cartridge-loading disc unit
Transfer Rate	92.6 KHz words/sec
Rotational Speed	1,500 rpm

continued

Capacity per Cartridge	1.17 million words
Track Access Time	110 milliseconds (maximum), 15 milliseconds (minimum), 60 milliseconds (average)
Dimensions	19 inches (48.7 cm) wide, 19.25 inches (45.5 cm) high
Weight	Approximately 100 pounds (45.5 kg)
Primary Power	115V ac (60 Hz); 230V ac (50 Hz)
Operational Environment	65 to 80 degrees F; 20 to 95 percent relative humidity without condensation
Disc Controller	
Control Capability	Four Model 620-37 disc memory units
I/O Capability	Ten external control and seven program sense instructions
Standard Device Address	014 through 017
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card
Interconnection	Interfaces with the computer and BIC via a 122-terminal connector and two 44-terminal connectors
Input Power	+5V dc at 2 amperes
Operational Environment	0 to 50 degrees C; 10 to 90 percent relative humidity without condensation

Table 9-11. Disc Memory Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 001x*	10001x	Select read mode and connect to BIC
EXC 011x	10011x	Select write mode and connect to BIC
EXC 021x**	10021x	Select and move to disc A
EXC 031x**	10031x	Select and move to disc B

continued

PERIPHERALS AND I/O INTERFACES

EXC 041x**	10041x	Select disc A for data
EXC 051x**	10051x	Select disc B for data
EXC 061x**	10061x	Enable interrupts
EXC 071x**	10071x	Initialize discs A and B

Transfer

OAR 011x	10311x	Transfer A register contents as starting sector address (or change head**)
OBR 021x	10321x	Transfer B register contents as starting sector address (or change head**)
OME 001x	10301x	Transfer memory contents as starting sector address (or change head**)

Program Sense

SEN 001x	10101x	Disc or controller failure
SEN 011x	10111x	Disc ready to select mode (file is protected**)
SEN 021x	10121x	Disc is busy
SEN 031x	10131x	Illegal address (disc A head at track 0**)
SEN 041x	10141x	Parity error (disc B head at track 0**)
SEN 051x	10151x	Track timing error (disc A or B move complete**)
SEN 061x	10161x	Transfer timing error (parity/overwrite**)
SEN 071x**	10171x	Position error

* x = 4 for Model 620-38 and 6 for Model 620-37 discs.

** Applies to Model 620-37 disc memory system only.

Drum Memories

The drum memory system (Models 620-44 through -49) consists of a drum memory unit, a dc power supply, and a controller.

Two basic systems are available. The smaller of the two drums can be expanded in increments of 16 tracks up to a maximum of 128 tracks. The larger model can be expanded in increments of 64 tracks up

to a maximum of 512 tracks. Each track stores 1,920 sixteen-bit words, providing a maximum storage capacity of 983,040 words in a 512-track system and 245,760 words in a 128-track system. The drum-stored word comprises 16 data bits and one parity bit.

The drum controller circuit card contains data registers and timing and control logic to control one drum memory unit. The controller:

- a. Controls bit-serial data transfers between the controller and drum memory unit
- b. Monitors and detects parity errors during read-from-drum operations
- c. Controls operational mode and provides interface control between the computer and the drum memory unit

The drum memory system is under the control of the BIC. Data are transferred between the controller and the memory unit via the drum cable.

The drum memory system specifications are given in table 9-12, and table 9-13 lists the controller instructions. Refer also to the drum memory system manual (document number 98 A 9902 280).

Table 9-12. Drum Memory System Specifications

Parameter	Description
	Drum Memory Unit
Organization	One fixed head per track; 16 to 512 tracks with 1,920 words per track
Transfer Rate	1.91 million bits per second (60 Hz, 106 KHz words), 1.60 million bits per second (50 Hz)
Rotational Speed	3,450 rpm (60 Hz); 2,880 rpm (50 Hz)
Write-to-Read Recovery Time	Less than 30 microseconds
Dimensions:	
128 Tracks	13 inches (32.9 cm) high, 18 inches (45.5 cm) wide, 18 inches (45.5 cm) deep
512 Tracks	25 inches (63.3 cm) high, 18 inches (45.5 cm) wide, 23 inches (58.2 cm) deep
Weight	128-track unit: 95 pounds (43 kg) 512-track unit: 200 pounds (90.6 kg)
Primary Power	117V ac (50 Hz), 230V ac (60 Hz), single phase
Operational Environment	10 to 40 degrees C; 20 to 80 percent relative humidity without condensation

continued

PERIPHERALS AND I/O INTERFACES

	Drum Controller
Control Capability	One drum memory unit storing up to 983,040 words
I/O Capability	Two external control, three transfer, and seven program sense instructions
Operational Modes	Read block data transfer, write block data transfer
Standard Device Address	014
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card
Interconnection	Interfaces with the computer and BIC via a 122-terminal connector; interfaces with drum unit via two 44-terminal connectors
Input Power	+5V dc at 2 amperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-13. Drum Memory Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0014	100014	Select read mode and connect to BIC
EXC 0114	100114	Select write mode and connect to BIC
Transfer		
OAR 0114	103114	Transfer A register contents as starting drum address
OBR 0214	103214	Transfer B register contents as starting drum address
OME 0014	103014	Transfer memory contents as starting drum address

continued

Program Sense		
SEN 014	101014	Drum failure
SEN 0114	101114	Drum ready to select mode
SEN 0214	101214	Drum busy with block transfer
SEN 0314	101314	Illegal address
SEN 0414	101414	Read parity error
SEN 0514	101514	Track timing error
SEN 0614	101614	Transfer timing error

Line Printer Equipment

The buffered line printer system (Model 620-77) comprises a high-speed line printer and a peripheral controller.

This operational, self-contained system offers high performance and printing quality in a wide range of on-line computer output applications. The line printer can also be used off-line by incorporating the required interface control logic.

Significant features and characteristics of the line printer system are:

- Segmented buffered line storage of up to 132 six-bit characters
- Programmed line-space control using a standard Teletype paper tape format
- Printing speeds of 245 to 1,100 lines per minute, providing 63 graphic and one blank (ASCII) character codes
- High reliability and printout quality as a result of friction-free, one-piece hammer construction

The line printer system specifications are given in table 9-14, and table 9-15 lists the line printer controller instructions.

Table 9-14. Line Printer Equipment Specifications

Parameter	Description
Line Printer	
Character Data	Format: standard ASCII Codes: 63 graphic and one blank Characters per line: up to 132 Horizontal spacing: ten characters per inch Vertical spacing: six characters per inch

continued

PERIPHERALS AND I/O INTERFACES

Dynamic Characteristics	Printing speed: 245 lines per minute (132 columns) to 1,110 lines per minute (24 columns) Drum rotation: 1,760 rpm Primary power: 110 to 130V ac (60 Hz, single phase) Current requirement: 500 watts
Paper	Type: standard fanfold, edge-punched, single copy minimum 15-pound bond, multicopy (up to six parts) of 12-pound bond with shot carbon Size: 4 to 19 inches (8.1 to 48.1 cm) wide with 22 inches (55.7 cm) between folds
Ribbon	Type: vertically fed roll Size: 14 inches (33.4 cm) wide and 20 yards (18m) long
Dimensions	46 inches (96.3 cm) high, 48.5 inches (122.7 cm) wide, and 25 inches (63.3 cm) deep
Weight	Approximately 575 pounds (240.5 kg)
Operational Environment	10 to 45 degrees C; 10 to 90 percent relative humidity without condensation
Organization	Printer Controller Contains timing and control logic, select/deselect logic, word buffer, and drivers and receivers
Control Capability	One Model 620-77 line printer, six-bit words
I/O Capability	Two external control, three transfer, and two program sense instructions
Standard Device Address	035-036
Logic Levels:	
Internal	Positive logic: True: +2.5 to +5.0V dc False: 0.0 to +0.5V dc
I/O and B Cables	Negative logic: True: 0.0 to 0.5V dc False: +2.5 to +3.7V dc

continued

Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card
Interconnection	Interfaces with the computer via a 122-terminal connector; interfaces with the printer via two 44-terminal connectors
Input Power	+5V dc at 500 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-15. Line Printer Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 003x	10003x	Initialize printer
EXC 013x	10013x	Connect printer to BIC
Transfer		
OAR 013x	10313x	Transfer A register contents to printer buffer
OBR 023x	10323x	Transfer B register contents to printer buffer
OME 003x	10303x	Transfer memory contents to printer buffer
Program Sense		
SEN 003x	10103x	Printer ready
SEN 013x	10113x	Character buffer ready

where x = last digit of the device address.

Oscilloscope Display System

The oscilloscope display system (Model 620-73) provides a visual display of the computer output. The system includes an oscilloscope with a 5-inch (8.7 cm) rectangular cathode ray tube (CRT), reference power supply, interconnecting cables, and a controller card.

The oscilloscope controller converts digital data to analog values that drive the horizontal and vertical deflection plates of the oscilloscope. The oscilloscope unit is designed for X-Y presentation. One of the

digital-to-analog (D/A) converters in the controller drives the X axis; the other, the Y axis. The Z channel input turns the CRT beam on and off.

The oscilloscope display system is directly under program control. The display is programmed by outputting data to one of the two D/A converters. Portions of the display can be inhibited by the program.

The oscilloscope display system specifications are given in table 9-16, and table 9-17 lists the oscilloscope controller instructions.

Table 9-16. Oscilloscope Display System Specifications

Parameter	Description
Oscilloscope	
Organization	Portable monitor unit with self-contained power supply; 5-inch (8.7 cm) rectangular CRT
Inputs	X, Y, and Z axis signals
Dimensions	6 inches (15.2 cm) high, 8.5 inches (21.5 cm) wide, and 17.5 inches (44.3 cm) deep
Primary Power	90 to 136V ac (50 Hz), 18 to 240V ac (60 Hz)
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation
D/A Converters	
Organization	12-bit; bipolar output
Accuracy	Better than 0.1 percent
Output	Single-ended, ± 10 volts full scale at 10 milliamperes, short circuit

continued

Settling Time	10 microseconds (maximum)
Conversion Time	100 Hz
Controller	
Organization	Contains decoding logic, two D/A converters, Z axis controller, and voltage regulator
Control Capability	One Model 620-73 oscilloscope unit
I/O Capability	Four external control and six transfer instructions
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card
Interconnection	Interfaces with the computer via backplane wiring; interfaces with power supply via the backplane connector; X, Y, and Z axis signals routed to the oscilloscope via three interconnecting cables
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-17. Oscilloscope Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0056	100056	Initialize X axis
EXC 0057	100057	Initialize Y axis
EXC 0256	100256	Turn CRT on
EXC 0356	100356	Turn CRT off
Transfer		
OAR 0156	103156	Transfer A register contents to X axis
OBR 0256	103256	Transfer B register contents to X axis
OME 0056	103056	Transfer memory contents to X axis
OAR 0157	103157	Transfer A register contents to Y axis
OBR 0257	103257	Transfer B register contents to Y axis
OME 0057	103057	Transfer memory contents to Y axis

Punched Card Equipment

Card Reader

The card reader system (Model 620-25) reads data from 80-column punched cards and transfers the data to the computer. The system consists of a card reader and a controller.

The card reader controller provides a nonbuffered interface between the reader and the computer. It also provides the timing and control logic to effect the transfers.

The controller can transfer data to the computer under direct program control or under the supervision of the BIC.

The controller specifications are given in table 9-18, and table 9-19 lists its instructions. Refer also to the card reader controller manual (document number 98 A 9902 260).

Card Punch

The card punch system (Model 620-27) comprises a punch and a controller. The controller controls data transfers from the computer to the card punch. The system can be operated under CPU control and, optionally, BIC control.

The data buffer register in the controller stores the 12-bit data words from the CPU. The control section synchronizes the control punch operation.

Under program control, the controller senses the punch (ready or not busy) and transfers data to it. Under BIC control, the controller requests data and the BIC controls the transfer from the specified memory addresses. Transfer of the data continues until terminated by the BIC.

The controller specifications are given in table 9-20, and table 9-22 lists its instructions.

Table 9-18. Card Reader Controller Specifications

Parameter	Description
Organization	Contains input receivers and output drivers for I/O bus and BIC interfacing, and a sequence controller to transfer data from the reader to the computer
Control Capability	One punched card reader unit
I/O Capability	Two external control, five transfer, and five program sense instructions
Reader Characteristics	Operating speed: 300 cards per minute Character length: 12 bits Card type: standard 80-column read on a per-column basis (end feed) Transfer rate: 1,050 characters per second

continued

Standard Device Address	030
Logic Levels	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card
Input Power	+5V dc at 250 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-19. Card Reader Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0030	100030	Initialize reader
EXC 0230	100230	Read one card
Transfer		
INA 0130	102130	Transfer character to A register
INB 0230	102230	Transfer character to B register
IME 0030	102030	Transfer character to memory
CIA 0530	102530	Transfer character to cleared A register
CIB 0630	102630	Transfer character to cleared B register
Program Sense		
SEN 0030	101030	End of card
SEN 0130	101130	Character ready
SEN 0230	101230	Reader error
SEN 0330	101330	Hopper empty
SEN 0630	101630	Reader ready

Table 9-20. Card Punch Controller Specifications

Parameter	Description
Organization	Contains line drivers and receivers for I/O and punch cabling, punch output data buffer, and control logic
Control Capability	One card punch unit
I/O Capability	Two external control, three transfer, and one program sense instructions
Punch Characteristics	Character length: 12 bits Card type: standard 80-column punched on a per-column basis Transfer rate: 35 cards per min.
Standard Device Address	031
Logic Levels	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card
Input Power	+5V dc at 485 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-21. Card Punch Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0031	100031	Initialize and enable BIC
EXC 0131	100131	Connect punch to BIC
Transfer		
OAR 0131	103131	Transfer initial buffer address to BIC from A register
OBR 0231	103231	Transfer initial buffer address to BIC from B register
OME 0031	103031	Transfer initial buffer address to BIC from memory
Program Sense		
SEN 031	101031	Card punch busy

Digital Plotter

The digital plotter system (Model 620-72) consists of a plotter and a controller.

The plotter produces high-quality, ink-on-paper graphic presentations of computer output. The controller provides a fully buffered interface to permit operation of the plotter under program control or under the direction of the BIC.

The plotter uses digital commands from the computer to produce the plot or drawing on a 12-inch (30.3 cm) wide roll of paper. These commands actuate step motors to produce incremental movement of the pen with respect to the paper. One step motor controls movement in two

directions along the X axis, and a separate motor controls movement on the Y axis. X-axis movement is accomplished by rotating the paper drum in either the +X or -X direction. The pen is moved to the left or right to effect movement on the Y axis. Composite commands can be given to move the pen on two axes simultaneously. This results in pen movements in any of eight directions.

The plotter operates at up to 300 increments per second, with three optional increment sizes: 0.005 inch, 0.010 inch, and 0.1 millimeter. The increment size is specified when the equipment is ordered.

The digital plotter system specifications are given in table 9-22, and table 9-23 lists the digital plotter controller specifications.

Table 9-22. Digital Plotter System Specifications

Parameter	Description
Plotter	
Organization	One digital plotter using digital computer commands to actuate step motors that produce the plot
Increment Sizes	0.005 inch (0.01 cm); 0.010 inch (0.03 cm); 0.1 mm
Operating Speed	300 increments per second (maximum)
Plot Dimensions	X axis: 11 inches (27.8 cm) Y axis: 120 feet (36.4m)
Pen	Liquid ink or ballpoint
Primary Power	105 to 125V ac, single phase, 50/60 Hz, 1.5 amperes (maximum)
Dimensions	98 inches (248 cm) high, 18 inches (45.5 cm) wide, and 14.7 inches (35.2 cm) deep

continued

PERIPHERALS AND I/O INTERFACES

Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation
Organization	Controller Contains interface circuitry, control and timing logic, and a six-bit command buffer
Control Capability	One digital plotter unit
I/O Capability	One external control, three transfer, and one program sense instructions
Standard Device Address	032
Input Power	+5V dc at 380 milliamperes, -5V dc at 42 milliamperes, and -12V dc at 15 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-23. Digital Plotter Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 032	100032	Initialize and connect to BIC
Transfer		
OAR 0132	103132	Transfer A register contents to command buffer
OBR 0232	103232	Transfer B register contents to command buffer
OME 0032	103032	Transfer memory contents to command buffer
Program Sense		
SEN 0132	101132	Buffer ready

Buffered I/O Controller

The buffered I/O controller (Model 620-80) provides a self-contained, programmable hardware interface for general-purpose data processing.

The input and output buffer registers provide parallel-word data communications between the computer I/O bus and an external device. In addition to data-handling,

the output buffer register can be programmed to output discrete control signals to an external device.

The buffered I/O controller uses a user-supplied cable, up to 20 feet (6m) long, for communication with external devices.

The buffered I/O controller specifications are given in table 9-24, and table 9-25 lists the controller instructions.

Table 9-24. Buffered I/O Controller Specifications

Parameter	Description
Organization	Contains operation and function decoder, input and output buffer registers, eight sense input gates and eight variable-pulsewidth control gates, and interface drivers and receivers
Control Capability	Provides buffered data transmission to and from external devices and the computer
I/O Capability	One external control, eight transfer, and one program sense instructions
Standard Device Address	060 through 067
Current Load	Sensing line input: nominally 7-milliamperes source at 0 volt Buffered output register: nominally 36-milliamperes source at 0 volt
Current Drive	Up to 65-milliamperes sink at 0 volt
Logic Levels	Positive logic: True: +2.5 to +5.5V dc False: 0.0 to +0.5V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card

continued

PERIPHERALS AND I/O INTERFACES

Interconnection	Interfaces with the computer via a 122-terminal connector; interfaces with external devices via two 44-terminal connectors
Input Power	+5V dc at 1.0 amperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-25. Buffered I/O Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0x6z	100x6z	Output control pulse on line selected by x from controller addressed by z
Transfer		
OAR 016z	10316z	Load output buffer of controller addressed by z from A register
OBR 026z	10326z	Load output buffer of controller addressed by z from B register
OME 06z	10306z	Load output buffer of controller addressed by z from memory
INA 016z	10216z	Read input buffer of controller addressed by z into A register
INB 026z	10226z	Read input buffer of controller addressed by z into B register
IME 006z	10206z	Read input buffer of controller addressed by z into memory
CIA 056z	10256z	Read input buffer of controller addressed by z into cleared A register
CIB 066z	10266z	Read input buffer of controller addressed by z into cleared B register
Program Sense		
SEN 0x6z	101x6z	Test state of line selected by x from controller addressed by z

where x = discrete control/sense line (00-07), and z = last digit of the controller device address.

Relay I/O Module

The relay I/O module (Model 620-83) provides a general-purpose, relay-buffered data link between special external devices and the computer. This I/O interface option has the capability of 16 relay-buffered inputs (620-83-2), 16 mercury-wetted relay contact outputs (620-83-1), or combined input/output (620-83-3).

In the relay-buffered input configuration (620-83-2), input relay contacts are activated by voltages from the user's equipment through the 12V dc input relay coil. Series resistors for coil input voltages greater than 12V can be installed. An energized input relay coil closes a contact that is gated into a 16-bit flip-flop register that can be accessed by the computer with one of five transfer instructions. The register can be cleared with external

control instruction or by system reset. The flip-flops remain set after an initial relay input until they are cleared.

The 620-83-1 relay output module allows isolated parallel transfer of a 16-bit word from the computer via mercury-wetted relay contacts to the user's equipment. The 16-bit word is clocked into a flip-flop register by any of three transfer instructions. The register drives 16 discrete circuits that, in turn, drive the 12V relay coils closing the contacts.

The relays can be cleared by an external control instruction, transferring all-zeros data out, or by system reset. The relay contacts remain closed until the flip-flops are cleared.

The specifications of the relay I/O module are given in table 9-26, and table 9-27 lists its instructions.

Table 9-26. Relay I/O Module Specifications

Parameter	Description
Relay Type	Output: mercury-wetted reed Input: dry reed
Contact Rating	Output: 50 volt-amperes resistive; 3 amperes or 400 volts (maximum) Input: 12 volt-amperes resistive; 0.5 ampere or 200 volts (maximum)
Contact Responses	Output: 0.05 ohm (average) Input: 0.2 ohm (average)
Capacitance	Output: 1 picofarad plus 0.01 microfarad external Input: less than 1 picofarad

continued

PERIPHERALS AND I/O INTERFACES

Operating Time	Output: 2 milliseconds (average) Input: 1 millisecond (average including bounce)
Release Time	Output: 2 milliseconds (average) Input: 1 millisecond (average)
Coil Power Consumption	Output: 500 milliwatts (average) Input: 100 milliwatts (average)
Drive Requirement	Output: three TTL load (through an amplifier) Input: 10 volts at 6.5 milliamperes (minimum)
Rated Load Life	Output: 25 million operations Input: 100 million operations
Standard Device Address	074 through 077
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-27. Relay I/O Module Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 007x	10007x	Clear all outputs
EXC 017x	10017x	Clear all inputs
Transfer		
OAR 017x	10317x	Transfer A register contents to buffered relay output contacts
OBR 027x	10327x	Transfer B register contents to buffered relay output contacts
OME 07x	10307x	Transfer memory contents to buffered relay output contacts
INA 017x	10217x	Transfer relay-buffered data to A register
INB 027x	10227x	Transfer relay-buffered data to B register

continued

IME 007x	10207x	Transfer relay-buffered data to memory
CIA 057x	10257x	Transfer relay-buffered data to cleared A register
CIB 067x	10267x	Transfer relay-buffered data to cleared B register

Program Sense

SEN 007x	10107x	Contact closed
----------	--------	----------------

where x = last digit of the device address.

Digital I/O Controller

The digital I/O controller (Model 620-81) provides a programmed link between an external device and the computer. There are eight control and sensing lines to permit the user to initialize, implement iterative control sequences, synchronize otherwise asynchronous external devices, and monitor the operation of such devices.

The controller operates entirely under program control. The function code of an I/O instruction defines one of eight control or sensing lines. In response to an external control instruction, the digital I/O controller places a pulse on the selected output

control line. Similarly, it responds to a program sense instruction by testing the operational state of an external device via the true or false level applied to the selected sense-response line.

Implementation of the digital I/O controller requires a user-supplied cable, up to 20 feet (6m) long, for communication between the controller and associated external devices.

Specifications of the digital I/O controller are given in table 9-28, and table 9-29 lists the controller instructions. Refer also to the digital I/O controller manual (document number 98 A 9902 612).

Table 9-28. Digital I/O Controller Specifications

Parameter	Description
Organization	Contains address and function decoders, eight nonbuffered control pulse output gates, eight nonbuffered sense response input gates, and interface drivers and receivers
Control Capability	Provides nonbuffered programmed link between the computer and external devices

continued

PERIPHERALS AND I/O INTERFACES

I/O Capability	One external control and one program sense instructions
Standard Device Address	060 through 067
Current Drive	Control pulse output: 48 milliamperes at 0 volt Sense response input: 6.5 milliamperes at 0 volt
Control Pulsewidth	170 nanoseconds
Transition Time	5 nanoseconds (minimum)
Logic Levels	Positive logic: True: +2.5 to +5.5V dc False: 0.0 to +0.5V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Interfaces with the computer via a 122-terminal connector; interfaces with external devices via two 44-terminal connectors
Input Power	+5V dc at 250 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-29. Digital I/O Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0x6z	100x6z	Output control pulse on line selected by x from controller addressed by z
Program Sense		
SEN 0x6z	101x6z	Test state of line selected by x from controller addressed by z

where x = discrete control/sense line (00-07), and z = last digit of the digital I/O controller device address.

Analog/Digital Conversion Equipment

Analog Input Module

The Models 620-850 and -851 Analog Input Modules (AIM) convert multiplexed analog input signals to digital values for use in the computer. The basic AIM comprises an analog-to-digital converter (ADC), a 16-channel (single-ended or differential) multiplexer, and control logic. The Model 620-88 A/D power supply is a prerequisite.

The multiplexer accepts, samples, and holds high-level analog signals for conversion by the ADC. The basic multiplexer services 16 differential, or single-ended, channels. This basic configuration can be expanded to a maximum of 256 channels in 16-channel increments. The multiplexer selects channels for input to the ADC either sequentially or on a random basis.

The ADC, using successive approximations, converts analog signals passed from the multiplexer to equivalent 13-bit digital values at an effective throughput of 50,000 conversions per second. ADC conversions can be initiated by the computer, an internal programmable timer, or an external pulse.

The control logic implements the transfer of the converted data to the computer

under program control or under the direction of the optional PIM and BIC.

The specifications of the AIM are given in table 9-30, and table 9-31 lists the AIM instructions.

Analog Output Module

The Models 620-870 through -875 Analog Output Modules (AOM) convert digital values to their equivalent voltage output signals. The basic AOM comprises one or two digital-to-analog converters (DAC) with 10-, 12-, and 14-bit resolution and control and addressing logic for up to eight analog output channels. The Model 620-88 A/D power supply is a prerequisite.

The AOM system can be expanded to eight DACs per device address to a maximum of 64. All DACs are initialized to zero-volt output by a SYSTEM RESET. They are selected for output under program control, and the computer outputs data to the DAC without reselection for each word until another DAC is selected or the system is reinitialized.

The control and addressing logic implements the transfer of the converted data under program control or under the direction of the optional BIC.

The specifications of the AOM are given in table 9-32, and table 9-33 lists the AOM instructions.

Table 9-30. Analog Input Module Specification

Parameter	Description
Conversion	50,000 samples per second (maximum)
Resolution	13 bits
Conversion Accuracy	± 0.012 percent, $\pm 1/2$ LSB
Conversion Time	13 microseconds (maximum)
Full-Scale Range	± 10 volts
Temperature Coefficient	± 50 microvolts per degree C (maximum)
Warm-Up Time	Essentially zero
Multiplexer Accuracy	± 0.01 percent
Source Impedance	1 kilohm
Voltage Range	± 10 V dc at 100 milliamperes
Output Impedance	20 ohms
Prerequisite	One Model 620-88 A/D power supply
Standard Device Address	054 through 057
Logic Levels	Negative logic: True: 0.0 to +0.4V dc False: +2.5 to +5.0V dc
Size	Two 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit cards
Interconnection	Interfaces with the computer and BIC via two 122-terminal connectors
Input Power	+5V dc ± 5 percent at 2 amperes ± 15 V dc ± 3 percent at 165 milliamperes +20V dc ± 5 percent at 15 milliamperes -22V dc ± 5 percent at 5 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-31. Analog Input Module Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0160	100160	Start conversion cycle
EXC 0240	100240	Select sequential scan of channels
EXC 0140	100140	Select random scan of channels
EXC 0040	100040	Set utility flip-flop output to ground
EXC 0340	100340	Reset utility flip-flop output to +5V dc
Transfer		
OAR 0160	103160	Transfer number of microseconds in timed interval
OAR 0140	103140	Transfer channel number
CIA 0560	102560	Transfer data to cleared A register
Program Sense		
SEN 0060	101060	ADC ready
SEN 0260	101260	Utility input
SEN 0160	101160	Timer flag
SEN 0061	10161	End of sequential scan

Table 9-32. Analog Output Module Specifications

Parameter	Description
Resolution	10, 12, or 14 bits
Conversion Accuracy	10 bits: ± 0.05 percent 12 bits: ± 0.012 percent 14 bits: ± 0.003 percent
Voltage Range	10 bits: $\pm 10V$ dc at ± 5 milliamperes 12 bits: $\pm 10V$ dc at ± 10 milliamperes 14 bits: $\pm 10V$ dc at ± 10 milliamperes
Temperature Coefficient	± 0.1 LSB per degree C
Warm-Up Time	Essentially zero

continued

PERIPHERALS AND I/O INTERFACES

Slew Rate	0.5V per microsecond, and 6V per microsecond
Settling Time	20 microseconds to stated accuracy
Adjustments	10 bits: full scale and zero 12 bits: full scale, zero, and MSB 14 bits: full scale, zero, and three MSB
Standard Device Address	050 through 053
Logic Levels	Negative logic: True: 0.0 to +0.4V dc False: +2.5 to +5.0V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Interfaces with the computer and BIC via a 122-terminal connector
Input Power	+5V dc ± 5 percent at 500 milliamperes +15V dc ± 0.1 percent at 50 milliamperes -15V dc ± 0.1 percent at 50 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-33. Analog Output Module Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0x5y	100x5y	Select channel x
Transfer		
OAR 0x5y	103x5y	Transfer A register contents to selected channel
Program Sense		
SEN 0x5y	101x5y	Status of line x
where x = channel (00 through 07), and y = last digit of the device address.		

Data Communications Equipment

The term *data communications* implies the transmission of digital information between two distant points. As presently used, the term also implies that two or more different transmission techniques are involved and that the data, therefore, exist in different forms during its transmission. The essential role of the equipment and interfaces described below is to act as translators and transmitters. A data communications interface must be able to handle data rates that can range from the hunt-and-peck keyboarding of a novice to the thousands of bits per second being transferred over high-speed telephone lines. It must be equally adaptable to the varying codes generated by Teletype, paper tape, magnetic tape, and punched card sources, all without losing a single bit of information in the process.

Three modes of communication are available. Depending on the application, they can apply to all types of communications services.

A *full-duplex* communications line can simultaneously carry information in both directions. A *half-duplex* line carry information in both directions, but not simultaneously. A *simplex* line is designed to carry data in only one direction; it is either transmitting or receiving.

In addition, there are two methods of transmitting data on a communications line. *Synchronous transmission* provides the highest data transfer rate for a given line capacity, but this is balanced by the increased costs and complexity of the transmitting and receiving equipment. *Asynchronous transmission* is slower, but makes use of simpler equipment. In synchronous transmissions, large blocks of

characters are transmitted as a continuous series of bits; in asynchronous transmissions, each character is transmitted as a unit with distinguishing start and stop signals identifying the beginning and end of the character.

Another important phase of data communications is the ability to *multiplex* several communications lines. Multiplexing is the process of transferring data from several storage devices operating at relatively low transfer rates to one storage device (e.g., a time-shared computer) operating at a high transfer rate. It is, therefore, the simultaneous transmission of a number of different messages over a single circuit.

Dataset Controllers

The dataset controllers (Models 620-65 and -66) interface with the computer and modems that are compatible with standard datasets. These controllers can operate in half- or full-duplex mode. They detect and establish input synchronization and switch to word mode for data transfers.

Data are transferred in one of three ways:

- a. The controller can be connected to the BIC for operations requiring minimum program intervention. BIC can be connected for half-duplex operation to input or output data. In full-duplex operation, I/O functions can be connected to the BIC and data flow can be controlled by the stored program.
- b. Full- or half-duplex operation can be completely controlled by the program.
- c. Inputting data to the PIM provides interrupt-controlled transfers.

PERIPHERALS AND I/O INTERFACES

The controller functions are:

- a. Receiving data from the modem and transferring it to the computer.
- b. Transmitting data received from the computer to the modem.

- c. Simultaneously receiving/transmitting data in both directions.

The Model 620-65 dataset controller specifications are given in table 9-34, and table 9-35 lists its instructions. Specifications for the Model 620-66 controller are given in table 9-36, and its instructions are listed in table 9-37.

Table 9-34. Model 620-65 Dataset Controller Specifications

Parameter	Description
Organization	Contains control logic, timing circuits, read and write buffers, and I/O drivers and receivers
Control Capability	One or two Bell 201 and 301, or equivalent, datasets
I/O Capability	Five external control, eight transfer, and four program sense instructions
Operational Modes	Full- and half-duplex, synchronous, automatic answer
Transfer Rate	2,000 to 2,400 bits per second, fixed rate
Standard Device Address	070 through 073
Logic Levels	Positive logic: True: +2.5 to +5.0V dc False: 0.0 to +0.5V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Interfaces with dataset via a 10-foot (3m) cable; interfaces with computer and BIC via the backplane wiring
Input Power	+5V dc at .6 amperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-35. Model 620-65 Dataset Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0071	100071	Initialize for search
EXC 0171	100171	Connect write buffer to BIC
EXC 0271	100271	Connect read buffer to BIC
EXC 0471	100471	Turn on request to send
EXC 0671	100671	Initialize for character format
Transfer		
OAR 0171	103171	Transfer eight LSB of A register to write buffer
OBR 0271	103271	Transfer eight LSB of B register to write buffer
OME 0071	103071	Transfer eight LSB of memory to write buffer
INA 0171	102171	Transfer read buffer contents to eight LSB of A register
INB 0271	102271	Transfer read buffer contents to eight LSB of B register
IME 0071	102071	Transfer read buffer contents to eight LSB of memory
CIA 0571	102571	Transfer read buffer contents to cleared eight LSB of A register
CIB 0671	102671	Transfer read buffer contents to cleared eight LSB of B register
Program Sense		
SEN 0171	101171	Write buffer empty
SEN 0271	101271	Read buffer full
SEN 0371	101371	Carrier on
SEN 0471	101471	Clear to send

Table 9-36. Model 620-66 Dataset Controller Specifications

Parameter	Description
Organization	Contains timing circuits, read and write buffers, modem control register, control logic, and I/O drivers and receivers

continued

PERIPHERALS AND I/O INTERFACES

Control Capability	One or two Model 33 or 35 Teletypes, or one or two Bell 103, or equivalent, datasets
I/O Capability	Two external control, eight transfer, and five program sense instructions
Operational Modes	Full- or half-duplex, asynchronous, automatic answer
Transfer Rate	110 to 300 bits per second
Standard Device Address	070 through 073
Logic Levels	Positive logic: True: +2.5 to +5.0V dc False: 0.0 to +0.5V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Interfaces with dataset via a 10-foot (3m) cable; interfaces with computer and BIC via the backplane wiring
Input Power	+5V dc at 0.7 amperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-37. Model 620-66 Dataset Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0471	100471	Initialize
EXC 0271	100271	Select load MCR
Transfer		
OAR 0171	103171	Transfer A register contents to write or MCR buffer
OBR 0271	103271	Transfer B register contents to write or MCR buffer
OME 0071	103071	Transfer memory contents to write or MCR buffer

continued

INA 0171	102171	Transfer read buffer contents to A register
INB 0271	102271	Transfer read buffer contents to B register
IME 0071	102071	Transfer read buffer contents to memory
CIA 0571	102571	Transfer read buffer contents to cleared A register
CIB 0671	102671	Transfer read buffer contents to cleared B register

Program Sense

SEN 0171	101171	Write buffer ready
SEN 0271	101271	Read buffer ready
SEN 0371	101371	Call connected
SEN 0471	101471	Call disconnected
SEN 0571	101571	Carrier on

Automatic Call Unit Controller

The automatic call unit controller (Model 620-62) interfaces between the computer and a Western Electric 801 Automatic Call Unit (ACU) to permit the computer to initiate the automatic dialing of any telephone number in a communications network. When such a call is acknowledged, the controller switches the line to a dataset for fully automatic transmission of data. The ACU thus performs all the functions of an attendant in originating a data call and transmission.

Telephone numbers to be called are included in the program stored in the computer and transferred to the ACU via the controller in four-bit, parallel configurations.

Operation of the ACU controller is under program control or under the direction of the optional BIC.

The ACU controller specifications are given in table 9-38, and table 9-39 lists its instructions.

Table 9-38. ACU Controller Specifications

Parameter	Description
Organization	Contains a four-bit output buffer, instruction decoder, and I/O drivers and receivers
Control Capability	One Western Electric 801 ACU
I/O Capability	Four external control, three transfer, and five program sense instructions

continued

PERIPHERALS AND I/O INTERFACES

Data Format	Four-bit-parallel; output meets RS232 specifications
Standard Device Address	070 through 073
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Input Power	+5V dc at 180 milliamperes -12V dc at 9 milliamperes +12V dc at 48 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-39. ACU Controller Instructions

Mnemonic	Octal Code	Description
ExternalControl		
EXC 007x	10007x	Initialize
EXC 017x	10017x	Enable call request
EXC 027x	10027x	Disable call request
EXC 047x	10047x	Reset digit present (DPR) flip-flop
Transfer		
OAR 017x	10317x	Transfer A register contents to controller output buffer
OBR 027x	10327x	Transfer B register contents to controller output buffer
OME 007x	10307x	Transfer memory contents to controller output buffer
Program Sense		
SEN 007x	10107x	Dial line busy
SEN 017x	10117x	Power indication
SEN 027x	10127x	Abandon call retry
SEN 037x	10137x	Call-origin status
SEN 047x	10147x	Next digit present

where x = last digit of the device address.

Data Communications Controller System

The Model 620-60 Data Communications Controller System (DCC-1) links the computer to remote Teletype terminals. The basic system consists of up to four 33/35 ASR Teletypes, 103A-type dataset modem, Teletype coupler, multiplexer, line controller, and power supply (Model 602-95-5).

Using telephone lines as the communications media, the DCC-1 implements the bidirectional transmission of binary serial data. It offers an economical, efficient means for on-line computer services to remote users. Typical applications include information storage and retrieval, classroom instruction, laboratory simulation, and statistical inquiry.

The DCC-1 multiplexer selects and controls the operation of the bidirectional data communications channels.

The line controller provides the interface between the computer I/O bus and the data communications channels.

A fully expanded DCC-1 can, under program control, service 16 full-duplex communications channels. The channels are serially interrogated and serviced on a request basis. System operation is controlled by predetermined program interrupts.

The DCC-1 controller specifications are given in table 9-40, and table 9-41 lists the system instructions.

Table 9-40. Data Communications Controller Specifications

Parameter	Description
Organization	Line controller: contains address gates, clock selection logic, and I/O drivers and receivers Multiplexer: contains sense and function control logic, address decoder and clock logic, interrupt decode/enable logic, and timing logic
I/O Capability	Eight external control and two program sense instructions
Standard Device Address	070 through 073
Logic Levels	Positive logic: True: +2.5 to +5.5V dc False: 0.0 to +0.5V dc
Size	Line controller: one 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card Multiplexer: one 7-3/4-by-12-inch (19.7 x 30.3 cm) wired-socket card

continued

PERIPHERALS AND I/O INTERFACES

Interconnection	Interfaces with the computer via a 122-terminal connector; interfaces with external devices via two 44-terminal connectors
Input Power	Line controller: +5V dc at 300 milliamperes Multiplexer: +5V dc at 600 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-41. Data Communications Controller Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 007x	10007x	Transmit a space
EXC 017x	10017x	Transmit a mark
EXC 027x	10027x	Disable communications channel
EXC 037x	10037x	Enable communications channel
EXC 047x	10047x	Select low-speed clock
EXC 057x	10057x	Select high-speed clock
EXC 067x	10067x	Disable interrupts
EXC 077x	10077x	Enable interrupts
Program Sense		
SEN 007x	10107x	Channel sense sample
SEN 017x	10117x	Channel sense data

where x = last digit of the device address.

Varian 620/DC System

The Varian 620/DC Data Communications System is a computer-controlled, time-multiplexed hardware/software system that can serve either as a data concentrator linking a number of low-speed communications lines to one or more high-speed lines, or as a preprocessor organizing incoming data from communications lines for direct entry into a large computer for final processing.

The Varian 620/DC adapts to a variety of communications network characteristics: widely varying transfer rates, synchronous/asynchronous transmissions modes, and full-duplex, half-duplex, and simplex operation. Essentially, the Varian 620/DC acts as the interface between the data sources, destinations, and transmission lines that comprise a data communications network, translating and transferring the information efficiently and accurately.

The major functional components of the Varian 620/DC are:

- a. The **Varian 620-Series Computer Memory** serves as a data reservoir between low- and high-speed communications lines, accepting and transmitting complete character and message blocks at line-compatible transfer rates.
- b. The **Model 620-68 Data Communications Controller (DCC-2)** stores control information for up to 64 communications lines. The DCC-2 assembles and disassembles data characters, acting as a buffer between the serial data transmitted by the lines and the

parallel data processed by the computer.

- c. The **Line Control Modules (Models 620-68-1 through -68-5)** interface the system signal levels and the levels of the modems connected to the communications lines.

Specifications of the Varian 620/DC are given in table 9-42, and table 9-43 lists the applicable instructions. Refer also to the Varian 620/DC manual (document number 98 A 9902 240), which describes installation and interfacing, gives a detailed theory of operation, and presents testing and troubleshooting procedures for maintaining the system.

Table 9-42. Varian 620/DC Specifications

Parameter	Description
Line Configuration:	
Capacity	Four (minimum) to 64 (maximum) communications channels for each Model 620-68 DCC-2
Expansion	Plug-in-expandable in four-channel increments to maximum line capacity (relay line control modules added in eight-channel increments)
Dynamic Characteristics:	
Operational Modes	Half-duplex, full-duplex, and simplex on any channel (selected under software control)
Transmission Modes	Synchronous at from 45 to 4,800 baud Asynchronous at from 45 to 1,200 baud
Programming Features	
Parity	Checked on incoming, and generated for outgoing, data lines, either odd or even
Priority	Lines can be designated high-priority under software control

continued

PERIPHERALS AND I/O INTERFACES

Line Interfaces:

Structure

Line control modules are plug-in-compatible with Western Electric 103, 201, 202, or commercial equivalent modems; other system components are compatible with asynchronous two-wire, asynchronous relay, and synchronous ground-isolated systems

Interconnection

Modem cables up to 50 feet (15m)

Installation

Mounts in standard 19-inch (48.1 cm) RETMA cabinet

Primary Power

115V ac ± 10 percent, 60 ± 2 Hz, single phase, up to 25 amperes

Operational Environment

0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 9-43. Varian 620/DC Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0170	100170	Enable interface request
EXC 0270	100270	Reset line control word
EXC 0370	100370	Read working register
EXC 0470	100470	System reset
EXC 0570	100570	Clear interface to not busy
EXC 0670	100670	Enable interrupts
EXC 0770	100770	Disable interrupts
EXC 0071	100071	Inhibit output interrupts until an input interrupt
EXC 0171	100171	Inhibit output interrupts
EXC 0271	100271	Output line break (all spaces)
EXC 0371	100371	Output mark character
EXC 0471	100471	Call character disassembly buffer
EXC 0571	100571	Call sequence controller
EXC 0671	100671	Call character assembly buffer
EXC 0771	100771	Call line identification field

continued

Transfer

OAR 0170	103170	Normal transfer of A register contents to interface register
OBR 0270	103270	Normal transfer of B register contents to interface register
OME 0070	103070	Normal transfer of memory contents to interface register
OAR 0171	103171	Mirror-image transfer of A register contents to interface register
OBR 0271	103271	Mirror-image transfer of B register contents to interface register
OME 0071	103071	Mirror-image transfer of memory contents to interface register
INA 0170	102170	Normal transfer of interface register contents to A register
INB 0270	102270	Normal transfer of interface register contents to B register
IME 0070	102070	Normal transfer of interface register contents to memory
INA 0171	102171	Mirror-image transfer of interface register contents to A register
INB 0271	102271	Mirror-image transfer of interface register contents to B register
IME 0071	102071	Mirror-image transfer of interface register contents to memory
CIA 0570	102570	Normal transfer of interface register contents to cleared A register
CIB 0670	102670	Normal transfer of interface register contents to cleared B register
CIA 0571	102571	Mirror-image transfer of interface register contents to cleared A register
CIB 0671	102671	Mirror-image transfer of interface register contents to cleared B register

Universal Controller

The universal asynchronous serial controller (Model 620-82) designed for interfacing any model 620 family computer to peripheral devices utilizing an asynchronous serial interface. This asynchronous controller should not be confused with a modem controller as it does not include modem control logic. The controller is intended for "direct connect" (no modem) interfacing of peripheral devices equipped with asynchronous serial interfaces. There are three versions of model 620-82:

Model No.	Interface Type
620-82A	RS-232C
620-82B	20 to 60 ma discrete current loop
620-82C	20 or 60 ma relay contact

Functionally, all three versions are the same. They differ only in their electrical interface characteristics; hence, selection of the proper model will depend upon the electrical interface requirement of the peripheral device to be connected.

Typical applications may be as follows:

- In-house teletypes equipped with RS-232C data set coupler (no modem required) — Version A.
- In-house or remote teletypes with 20 or 60 ma relay or discrete current loop interface — Version B or C.
- CRT displays.
- Computer to computer links.
- Certain serial printers, cassettes, and other serially interfaced devices.
- 83 B 1 equipment.

All versions of the controller are serial, direct connect, character buffered units capable of half or full duplex operation. Modem control logic is not included. The existing TTY command set is employed.

Operation can be either under program control or in the interrupt mode using the Priority Interrupt Module (PIM) option. Also, automatic block transfers are possible when the operation is in conjunction with the Buffer Interface Controller (BIC) option. Table 9-44 depicts specific characteristics of each version.

All versions are tested in a back-to-back manner, i.e., CPU output data is "wrapped-around", input back to the CPU and a comparison made for proper transmission and reception. Test programs for checkout of specific peripherals and connection to the peripheral device are the user's responsibility, except in the cases where VDM also provides the peripheral device. When VDM supplies the controller and peripheral device, a test routine is provided to exercise the functions of the peripheral device connected to the controller.

The standard 20 ft. cable kit furnished with each controller consists of two twisted pairs (24 gauge) in an overall jacket.

A controller mating connector is supplied in the kit. Cables longer than 20 ft. are optionally available and must be specified when ordering. The kit also contains material for a test connector.

The controller is packaged on a printed circuit board, has rear edge connectors and occupies one I/O slot in the mainframe or I/O expansion chassis.

The operating distance on model 620-82A is guaranteed to 50 ft.; in most cases, operation up to 100 feet is feasible. Users requiring operating distances beyond 100 ft. should use model 620-82B. Maximum operating distance for models 620-82B and 620-82C depends upon the serial bit rate. Operation at teletype rate, for example, can be up to 10,000 ft. The following is a

guideline on maximum distance versus BPS rate under normal conditions.

Distance up to 1,000 ft.
10,000 BPS maximum
Distance up to 2,000 ft.
4,800 BPS maximum
Distance up to 3,500 ft.
1,800 BPS maximum
Distance up to 5,000 ft.
900 BPS maximum
Distance up to 10,000 ft.
300 BPS maximum

Standard Baud rates (within the above limits are): 45, 75, 110, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600.

The user should specify the serial rate, character size, stop bits and cable length. When variable data is not specified, the controller will be equipped as follows:

- 620-82A/620-82B
1200 BPS; 8 level; one stop bit; 20 ft. open-ended cable.
- 620-82C
110 BPS; 8 level; two stop bits; 20 ma; 20 ft. open-ended cable.

When either model 620-82B or 620-82C is used, the peripheral device must have a current loop interface. This can be of the relay type or a discrete circuit similar to the circuit on the controller. The user is responsible for providing the "line battery" source for the current loops. The line battery source typically can be obtained from the peripheral device or a small separate power supply. Any voltage from 12VDC to 50VDC can be used.

Table 9-44 lists specific characteristics for each version of the controller.

Table 9-44. Universal Controller Specifications

Parameter	Description
Model Number 620-82A	
Serial Rate, BPS (Bits per second)	45 to 9,600 BPS depending upon cable length (see Table 2).
Character Size	5, 6, 7, or 8 level (bits). Parity bit (if any) is added as one of the bits — i.e., 7 bits and parity = 8 bits; or 8 bits and no parity = 8 bits.
Stop Bit	One or two
Cable and Distance	Standard length = 20 ft., optionally up to 100 ft.
Model Number 620-82B	
Serial Rate, BPS (Bits per second)	Same as 620-82A
Character Size	Same as 620-82A
Stop Bit	Same as 620-82A
Cable and Distance	Standard length = 20 ft., optionally up to one mile

continued

Miscellaneous

User must provide "line battery" for loop current.

Model Number 620-82C

Serial Rate, BPS (Bits per second)

45 to 300 BPS. Typically would be 75 or 110 BPS. Relays will not operate reliably at higher than 300 BPS

Character Size

Same as 620-82A

Stop Bit

Same as 620-82A

Cable and Distance

Same as 620-82B

Miscellaneous

Same as 620-82B

Disc Memory Systems**Models 620-35 and -35A**

Model 620-35, -35A Disc Memory System is a peripheral mass storage option for Varian 620-100 computers. The system — which includes a disc drive unit, an interface controller capable of operating up to four disc units, a removable disc pack, and all required interconnecting cables — provides storage of up to 29.2 million eight-bit bytes

of data for each disc drive unit. In addition, the multiple-disc capability of the controller allows additional disc drives (up to four per controller) to be added at any time to increase the storage capability of the system. The "add-on" disc drive units are assigned model number 620-35A.

Specifications for the 620-35, -35A discs are given in Table 9-45. Disc memory controller instructions are listed in Table 9-46.

Table 9-45. Model 620-35, -35A Disc Memory Specifications

Parameter	Description
OEM Model Number	Century Data CDS-114
Storage Capacity	29.2 million eight-bit bytes per disc drive unit.
Recording Mode	Double Frequency
Data Transfer Rate	2.50 MHz (bit rate), 312,000 bytes per second. 156 K words/sec.
Rotation Speed	2400 RPM $\pm$ 2%
Average Latency Time	12.5 milliseconds
Track-to-Track Access Time	10 milliseconds
Maximum (Full-Stroke) Access Time	65 milliseconds
Input Power	208 VAC $\pm$ 10%, 3-phase, 4-wire, wye-connected, 60Hz (+1%, -2%), 1000 watts.
Motor Starting Current	25 Amperes for 4 seconds
Power Factor	0.8

continued

Heat Dissipation	3500 BTU/hour
Construction	Free-standing unit, 40" high, 30" wide, 24" deep
Weight	425 pounds
Space Requirements	3 feet clear at front and back for access to service panels.
Operating Environment	15.6° to 32.2°C (60° to 90°F) with a maximum gradient of 20°F per hour. 10% to 80% relative humidity (no condensation).

Table 9-46. Disc Memory Controller Instructions

Mnemonic	Octal Code	Description
External Control Instructions		
EXC 0015	100015	Stop transfer and initialize
EXC 0115	100115	Select cylinder (seek)
EXC 0215	100215	Read address
EXC 0315	100315	Write track format
EXC 0415	100415	Read/Write one record
EXC 0416	100515	Recalibrate, return heads to zero
Sense Instructions		
SEN 0015	101015	Sense disc drive 0 selecting a cylinder
SEN 0115	101115	Sense disc drive 1 selecting a cylinder
SEN 0215	101215	Sense disc drive 2 selecting a cylinder
SEN 0315	101315	Sense disc drive 3 selecting a cylinder
SEN 0415	101415	Sense DCU not busy
Transfer Instructions		
IME 0015	102015	Input to the 620/f-100 memory
INA 0115	102115	Input to A register
INB 0215	102215	Input to B register
CIA 0515	102515	Clear and input to A register
CIB 0615	102615	Clear and input to B register
OME 0015	103015	Output from the 620/f memory
OAR 0115	103115	Output from A register
OBR 0215	103215	Output from B register

Model 620-36

Model 620-36 Disc Memory System is a peripheral mass storage option for the Varian 620-100 family of computers. The system includes a disc drive unit with a storage capacity of 2,338,000 16-bit words, a disc power supply, an interface controller, and all required interconnecting cables and mounting hardware. The disc unit contains one non-removable rotating disc and a single-disc removable disc pack (IBM 2315 24 sector or equivalent).

The disc controller and disc power supply are both capable of operating either one or two disc drive units. Therefore, the mass storage capacity of the system can be doubled by addition of a slave disc drive unit (Model 620-36A).

Specifications for the 620-36, -36A are given in Table 9-47. Disc memory controller instructions are given in Table 9-48.

Table 9-47. Model 620-36, -36A Disc Memory Specifications

Parameter	Description
OEM Model Number	PERTEC Model 5221
Storage Capacity	2,338,000 16-bit words per disc drive unit
Recording Mode	Double Frequency
Data Transfer Rate	1.562 Mhz (bit rate), 92K words per second
Rotation Speed	1500 RPM
Average Access Time	60 milliseconds
Track-to-Track Access Time	15 milliseconds
Input Power	All operating power supplied by disc power supply.
Dimensions	19" wide, 17-1/2" high, 28-1/2" deep
Weight	approximately 100 pounds
Operating Environment	+18° to +35°C (+65° to +95°F), 0 to 80% relative humidity (no condensation).

Disc Power Supply

Capability	Supplies all operating voltages for two disc drive units
Input Power	115 VAC / 60 Hz @ 3 to 6 amps 230 VAC / 50 Hz @ 2 to 4 amps
Dimensions	19" wide, 8-3/4" high, 23-1/2" deep
Weight	100 pounds (maximum)
Operating Environment	Same as disc drive unit

Table 9-48. Disc Memory Controller Instructions

Mnemonic	Octal Code	Description
EXC Command		
EXC 0016	100016	Select Read Mode and Connect BIC
EXC 0116	100116	Select Write Mode and Connect BIC
EXC 0216	100216	Set Controller to Seek Mode
EXC 0316	100316	Set Controller to Sector Select Mode
EXC 0416	100416	Initialize Controller
EXC2 0016	104016	Select Disc Drive 0, Pack 0 (non-removable)
EXC2 0116	104116	Select Disc Drive 0, Pack 1 (removable)
EXC2 0216	104216	Select Disc Drive 1, Pack 0 (non-removable)
EXC2 0316	104316	Select Disc Drive 1, Pack 1 (removable)
SEN Commands		
SEN 0016	101016	Seek Complete — Disc 0, Pack 0
SEN 0116	101116	Seek Complete — Disc 0, Pack 1
SEN 0216	101216	Seek Complete — Disc 1, Pack 0
SEN 0316	101316	Seek Complete — Disc 1, Pack 1
SEN 0416	101416	Sense Controller Busy
SEN 0516	101516	Sense Error (see status word)
SEN 0616	101616	Sense Selected Disc Not Ready
SEN 0716	101716	Sense Selected Disc Write Protected
Data Transfer Commands		
OAR 0116	103X16	Transfer Out (Track/Sector Address)
CIA 0516	102X16	Transfer In (Status Word)

SECTION 10 - PHYSICAL CONFIGURATION

An outstanding feature of the Varian 620/f-100 is its compact design. Basic systems containing up to 16K of memory consist of a mainframe and memory chassis, and require 21 inches (53.3 cm) of vertical rack space. Systems containing 20K through 32K of memory consist of a mainframe, memory chassis, and an I/O expansion chassis, and require 31-1/2 inches (78.0 cm) of vertical rack space. The central processing unit (CPU), memory protection (MP), power failure/restart (PF/R), real-time clock (RTC), hardware multiply/divide (M/D), power supply, and computer options are housed in the mainframe. The memory chassis accommodates either 16K of memory and up to 13 I/O controller cards, or 4K through 32K of memory. Systems with 20K to 32K of memory include an I/O expansion chassis with 12 I/O card slots. An additional 12 I/O card slots can be added by adding another half backplane. Additional I/O expansion is available in all configurations by adding I/O expansion chassis, I/O wiring backplanes, and expansion power supplies if necessary.

System Planning

The types of chassis available with the Varian 620/f-100 computer system are:

- a. Mainframe
- b. Memory chassis
- c. I/O expansion chassis

Mainframe

The mainframe contains the control panel, CPU, MP, PF/R, M/D, RTC, power supply, and computer options. The computer options are:

- a. Teletype controller
- b. Automatic bootstrap loader
- c. Priority memory access

Memory Chassis

The memory chassis contains the memory, which can vary in size from 4K to 32K in 4K increments. The memory chassis can also accommodate peripheral controller cards (in addition to 16K of memory). Up to 13 etched controller cards, up to four wire-wrapped controller cards, or a combination of both can be installed in the memory chassis (one card slot contains an I/O termination shoe).

I/O Expansion Chassis

The I/O expansion chassis can accommodate up to 22 etched-circuit controller cards, up to eight wire-wrapped controller cards, or a combination of both. One card slot contains an I/O termination shoe and one card slot contains an I/O expansion cable connector.

Space Requirements

The 620/f-100 computer chassis are in individual cabinets suitable for standard

PHYSICAL CONFIGURATION

RETMA rack or table-top installation. The mainframe is 10.5 inches (26.6 cm) high, 19 inches (48.3 cm) wide, and 21 inches (53.1 cm) deep. The memory and I/O expansion chassis are 10.5 inches (26.6 cm) high, 19 inches (48.3 cm) deep, and 19 inches (48.3 cm) wide. The mainframe and memory chassis contain fans for cooling. Space must be provided for adequate air flow.

The standard 33 Teletype unit with stand is approximately 33 inches (83.5 cm) high, 19 inches (48.3 cm) deep, and 22 inches (55.7 cm) wide.

In standard 620/f-100 systems, the memory chassis is located directly below the mainframe, and the I/O expansion chassis must be located directly below the memory chassis. The dc power cable on the expansion power supply has a maximum length of 6 feet (1.8m). Deviations from the standard system configuration may exist as an optional feature to meet special user requirements.

Power Requirements

The mainframe power supply provides power for the CPU cards, 32K of memory, and all option cards. In addition, approximately 5 amperes are available for peripheral controller cards. If additional power for peripheral controllers is required, a 620 expansion power supply is available.

The mainframe and expansion power supplies connect to a standard 115-volt, 60 Hz power source. Also available, for European installations, are power supplies for use with 230-volt, 50 Hz sources.

These power supplies need not be regulated under normal commercial power conditions.

With maximum loads, the mainframe power supply draws approximately 8 amperes of ac power, and the expansion power supply, approximately 4 amperes.

Environmental Requirements

Ambient temperature at the installation site can vary between 0 to 50 degrees C with no adverse effects on computer operations. To extend the life expectancy of the computer, however, it is recommended that ambient temperature be maintained between 15 and 30 degrees C.

Relative humidity can be up to 90 percent (without condensation).

Physical Description

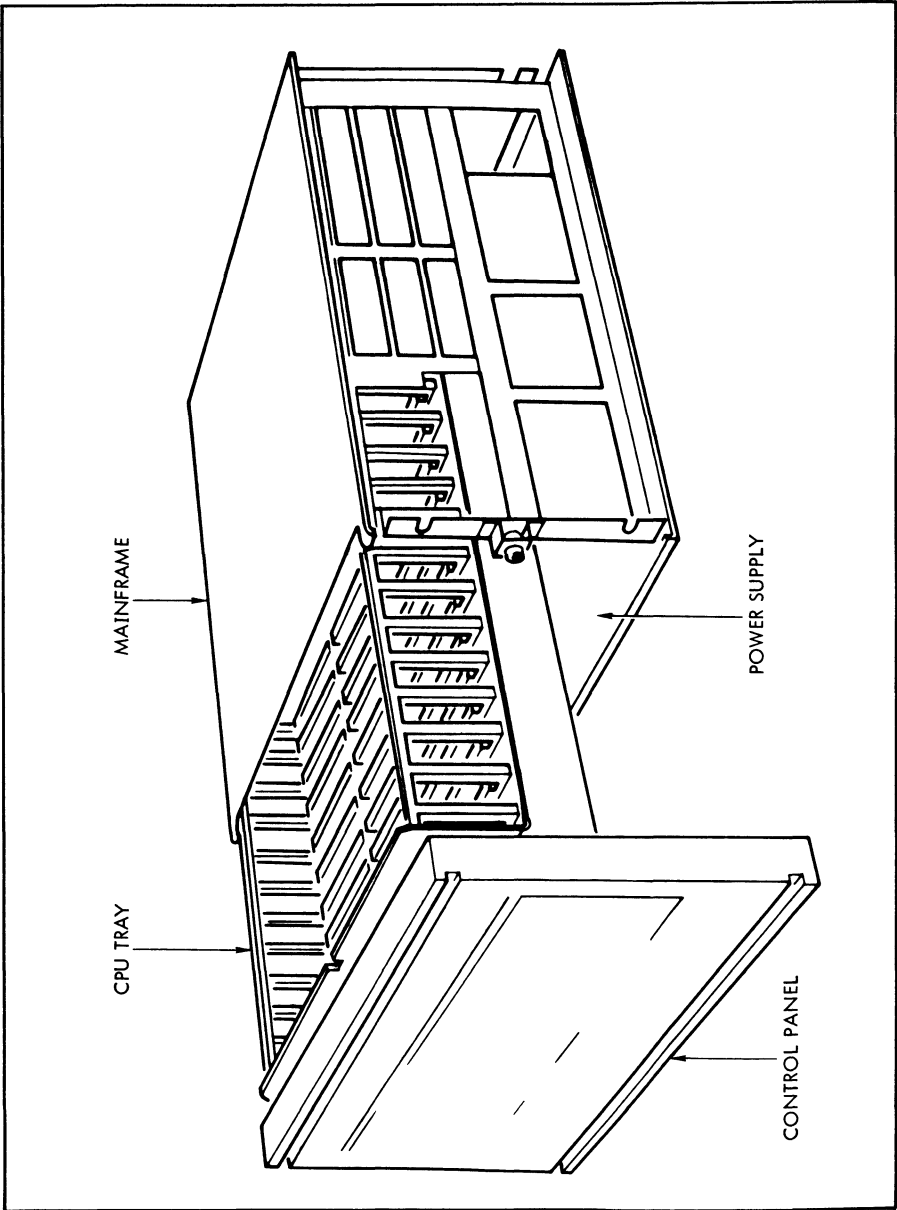
The 620/f-100 computer chassis are fabricated from cold-rolled steel with spot-welded construction. The steel is cadmium-plated per military specification QQ-P-416, type II, class 2, for corrosion protection.

The mainframe contains six fans mounted on the right side (when viewed from the front) to provide cooling for the electronic components. Fans in the memory chassis are located under the card slots.

Mainframe

The mainframe (figure 10-1) contains the control panel, CPU tray, and power supply. The molded-plastic control panel contains all the necessary controls and indicators to operate the computer. The display card is mounted behind the panel. This card holds the lamps and switches, and associated circuits. The lamp can be replaced without soldering.

The control panel is located on the front of the mainframe. To unlatch the panel,



VTII-1447

Figure 10-1. Mainframe With Control Panel Extended

PHYSICAL CONFIGURATION

press the pushbutton fastener on the left side. The front panel can then be pulled out to expose the CPU tray and the power supply.

As an optional feature, the control panel can also be installed up to 25 feet (7.6m) from the mainframe. In this case, the mainframe front panel is blank.

The display card connects to the CPU tray via two flat cables that connect to J46 and J47 on the CPU tray.

The CPU tray is located in the upper portion of the mainframe. As illustrated in figure 10-2, the CPU tray has 14 card slots to accommodate CPU and optional circuit cards. Table 10-1 lists the names and slot locations of the circuit cards in the CPU tray.

The display card connects to the CPU tray via two flat cables that connect to J46 and J47 on the CPU tray. A memory data cable connects between CPU tray connector J48 and connector J28 on the memory chassis. A memory address cable connects between CPU tray connector J52 and connector J27 on the memory chassis. A dc power cable from the power supply connects to CPU tray connector J50. To route I/O signals to the memory chassis, an I/O cable connects between CPU tray connector J49 and connector J29 on the memory chassis. To route PMA signals to a peripheral controller, a PMA cable connects between CPU tray connector J51 and the peripheral controller.

The CPU tray is mounted on slides and can be extended through the front of the mainframe for servicing. To extend the CPU tray:

- a. Release the pushbutton fastener on the control panel.
- b. Pull the CPU tray out to the desired position.

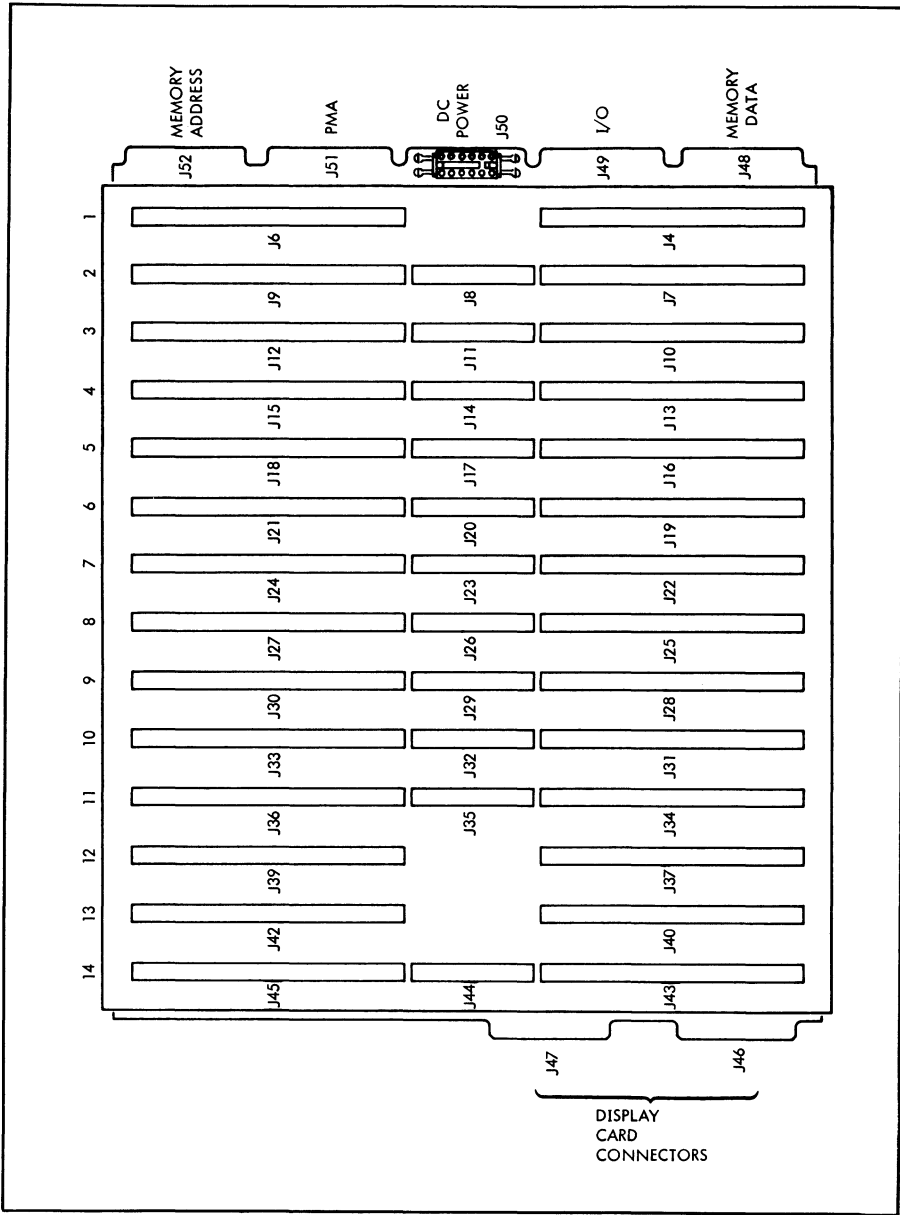
The CPU tray circuit cards are 14.9 inches (37.3 cm) long and 3.1 inches (7.8 cm) wide (figure 10-3).

To install a card, insert it into the mounting guides of the designated slot with the component side of the card toward the rear of the mainframe. Apply moderate pressure to the card, forcing the three card-edge connectors to seat firmly in their mating connectors in the CPU tray. Relieve the force from the extended CPU tray by placing fingers under the support bar. To remove a card, lift the inside edge of the ejector handles to unseat the card from the mating connectors then lift the card from the slot.

The computer power supply is located in the lower portion of the mainframe (figure 10-1). The power supply contains no fans, but is cooled from fans in the mainframe.

The front and rear panels of the power supply are illustrated in figure 10-4. The rear panel contains four screws that fasten it to the rear of the mainframe. The power cables, which are wired to the power supply, are routed through holes in the rear panel. The front panel contains an ac indicator and an ac circuit breaker. The ac indicator lights when ac line voltage is applied to the power supply. The ac circuit breaker is activated if a component failure within the supply causes excessive input current.

The power supply can be extended out through the rear of the mainframe after removing the four screws on the rear panel (figure 10-4). To remove the power supply from the mainframe, the computer power must be turned off and power cables disconnected.



VT11-0968A

Figure 10-2. Top View of CPU Tray

PHYSICAL CONFIGURATION

Table 10-1. Card Locations

Slot	Description	Part No.
1	I/O control	44P0442
2*	Priority memory access	44P0481
3	I/O data	44P0443
4	Optional instruction set	44P0444
5	Data loop 12-15	44P0448
6	Data loop 8-11	44P0447
7	Data loop 4-7	44P0446
8	Data loop 0-3	44P0445
9	Control I	44P0449
10	Control II	44P0450
11	Clock	44P0440
12	Memory protection	44P0480
13*	Teletype controller and automatic bootstrap loader	44P0451
14	Real-time clock and power failure/restart	44P0483

* Card slot for optional circuit card.

Memory Chassis

The memory chassis contains a front panel, a connector panel, circuit card slots, and a backplane (figure 10-5).

The front panel is blank. To unlatch the front panel, press the pushbutton fasteners on both sides. The panel can then be removed to expose the chassis wiring and the memory and I/O cables.

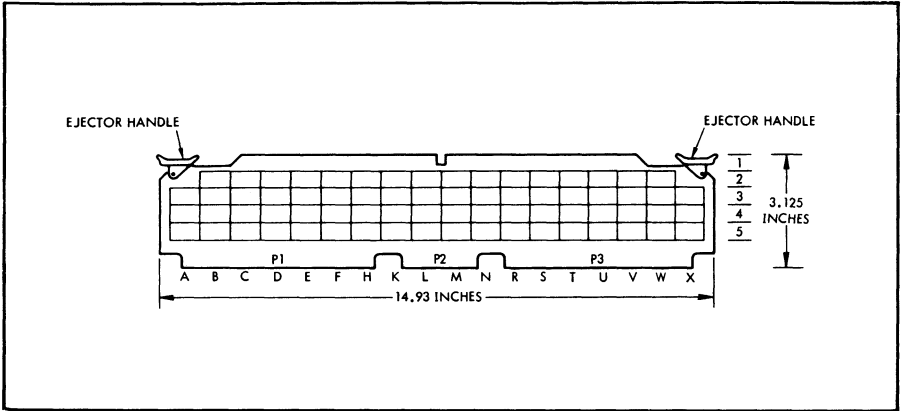
The connector panel (figure 10-6) is located at the rear of the memory expansion chassis and contains power connector J30 and Teletype connector J31.

The memory chassis can accommodate memory and peripheral controller cards. Except for the memory stack card (mounted on the sense/inhibit card), the

cards are 7-3/4-by-12 inches (19.7 by 30.3 cm), each containing a 122-pin connector for mating with a backplane connector in the memory chassis. The cards are installed through the rear of the memory chassis with the component side on the left (except the memory stack card). Card-slot assignments for all memory cards are listed in table 10-2.

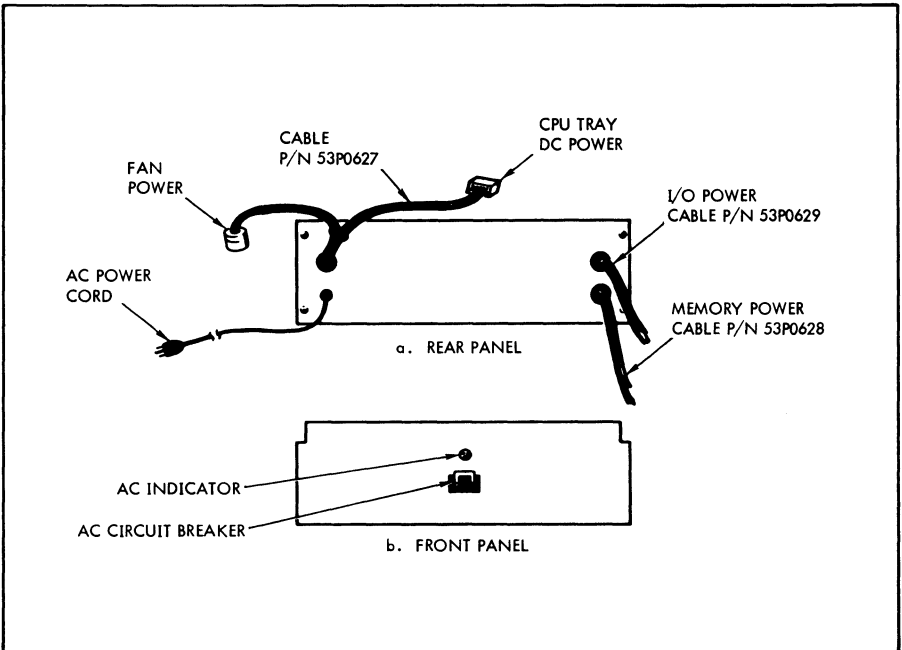
The card slots occupied by the various 4K increments of memory are:

1st 4K:	slots 1, 2, 3, 4, 5
2nd 4K:	slots 6, 7
3rd 4K:	slots 8, 9, 10
4th 4K:	slots 11, 12
5th 4K:	slots 13, 14, 15
6th 4K:	slots 16, 17
7th 4K:	slots 20, 21, 22
8th 4K:	slots 23, 24



VT11-0969

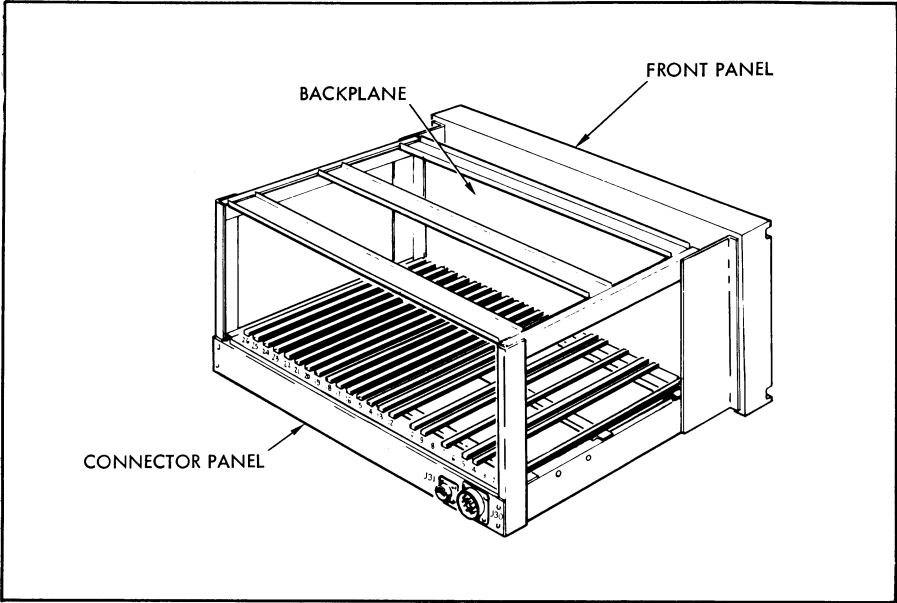
Figure 10-3. Typical Circuit Card in CPU Tray



VT11-1324

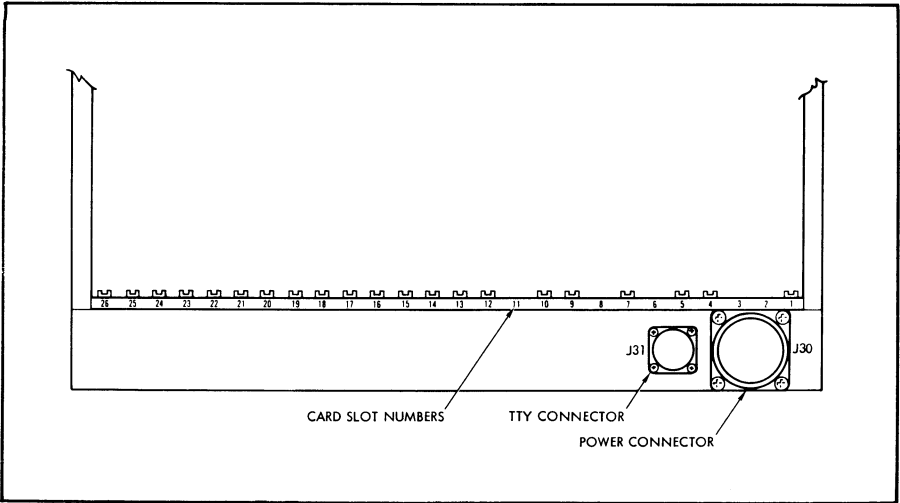
Figure 10-4. Power Supply Panels

PHYSICAL CONFIGURATION



VTII-1326

Figure 10-5. 16K Memory Chassis



VTII-1327

Figure 10-6. 16K Memory Chassis Connector Panel

Table 10-2. Memory Card Assignments

Slot	Name	Number
1	Memory timing, data and address reg	44P0579
3*	Memory stack	50S0012
4	Sense/inhibit	44P0506
5	Driver/sink switch	44P0578
6*	Memory stack	50S0012
7	Sense/inhibit	44P0506
8*	Memory stack	50S0012
9	Sense/inhibit	44P0506
10	Driver/sink switch	44P0578
11*	Memory stack	50S0012
12	Sense/inhibit	44P0506
13*	Memory stack	50S0012
14	Sense/inhibit	44P0506
15	Driver/sink switch	44P0578
16*	Memory stack	50S0012
17	Sense/inhibit	44P0506
20*	Memory stack	50S0012
21	Sense/inhibit	44P0506
22	Driver/sink switch	44P0578
23*	Memory stack	50S0012
24	Sense/inhibit	44P0506

* The memory stack card occupies this card slot, but does not plug into the slot connector; it plugs into a right-angle connector mounted on the sense/inhibit card.

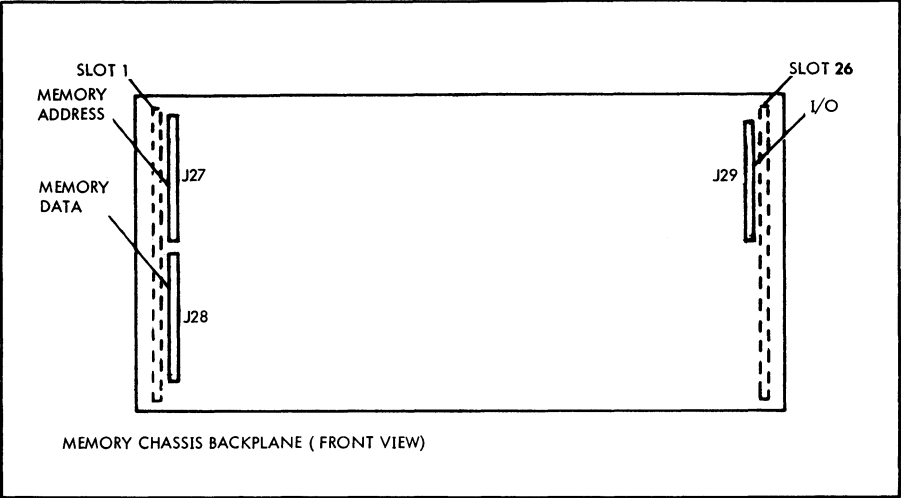
The memory chassis backplane contains circuit card connectors, cable connectors, and wiring for memory and/or I/O cards. There are two types of backplane wiring configurations available for the memory chassis. One backplane contains wiring for up to 16K of memory in card slots 1 through 12 and 13 etched controller cards (or four wire-wrapped controller cards) in card slots 13 through 26 (one card slot contains an I/O termination shoe). The other type of backplane contains wiring for up to 32K of memory in card slots 1 through 24. Three 50-pin cable connectors

(J27, J28, and J29) are mounted on the front of the backplane (figure 10-7). Connectors J27 and J28 are located between card slots 1 and 2; J27 connects to the memory address cable and J28 connects to the memory data cable. Connector J29 is located between card slots 25 and 26 and connects to the I/O cable.

I/O Expansion Chassis

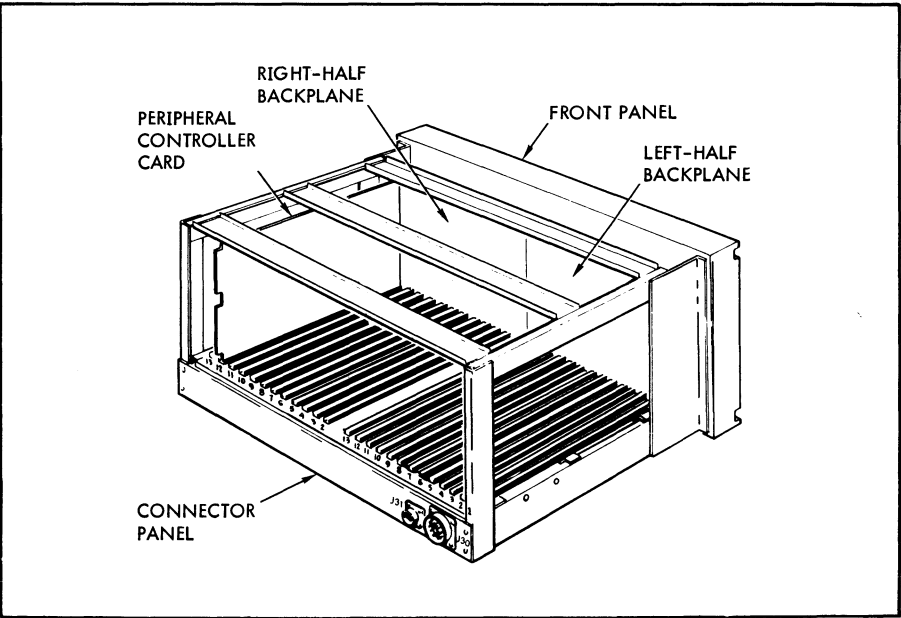
An I/O expansion chassis containing a peripheral controller card is illustrated in figure 10-8. The front panel is the same as

PHYSICAL CONFIGURATION



VTII-1328

Figure 10-7. Memory Chassis Cable Connectors



VTII-1329

Figure 10-8. I/O Expansion Chassis

that on the memory chassis. The connector panel, located at the rear of the chassis, contains expansion power supply connector J31 and mainframe power supply connector J30. The circuit card slots are numbered at the rear of the chassis from right to left. The left-half backplane contains card slots 2 through 13, and the right-half backplane, card slots 2 through 13.

The I/O expansion chassis can accommodate up to 22 etched controller cards, up to eight wire-wrapped controller cards, or a

combination of both. This chassis interconnects with the memory chassis via a flat I/O cable, which normally connects to slot 13 in the right-half backplane of the chassis. An I/O termination shoe must be installed in the last card slot.

Each peripheral controller card is 7-3/4-by-12 inches (19.7 by 30.3 cm), and contains a 122-pin connector for mating with a backplane connector in the I/O expansion chassis. The other end of the card contains two 44-pin connectors that mate with peripheral device cables.

SECTION 11 - SYSTEM INTERCONNECTION

Figure 11-1 identifies connectors used for interconnection of the mainframe, memory chassis, and I/O expansion chassis. The connectors are located at the rear of the chassis except for connectors J27, J28, and J29, which are located on the front of the memory chassis backplane.

Memory Interconnection

The memory chassis is connected to the mainframe with two flat memory cables and a flat I/O cable. The cables each contain 50 shielded wires. The routing of these cables is illustrated in figure 11-2. The memory address cable (part number 53P0619-040) connects between J52 on the rear of the CPU tray and J27 on the front of the memory chassis backplane. The memory data cable (part number 53P0619-051) connects between J48 on the rear of the CPU tray and J28 on the front of the memory chassis backplane. The I/O cable (part number 53P0619-042) connects between J49 on the rear of the CPU tray and J29 on the front of the memory chassis backplane. The length of each cable (in inches) is designated by the dash number in the cable part number.

I/O Expansion Interconnection

The I/O expansion chassis is connected to the memory chassis with a flat I/O expansion cable (part number 53P0547) that has a 122-pin card connector at both ends. This cable connects to card slot 13 in the I/O expansion chassis, and to card slot 13 or 26 in the memory chassis. Card slot 26 is used in systems that have a

memory size greater than 16K (figure 11-3). Card slot 13 is used in systems that have a memory size of 16K or less (figure 11-4). An I/O termination shoe (part number 44P0530) must be installed in the last I/O card slot of the system.

Power Supply Interconnection

The mainframe power supply (part number 83P0041) provides power for the CPU, 32K of memory, all internal options, and a limited number of peripheral controllers (up to 5 amperes).

The power cable connections for a system without a Varian 620 expansion power supply are illustrated in figure 11-5. The Y-type CPU power cable (part number 53P0627) routes power to J50 on the CPU tray and J1 on the fan panel assembly. The memory power cable (part number 53P0628) is routed to J30 on the memory chassis to provide power for the memory cards, peripheral controller cards, and fans. If controller cards requiring voltages other than +5 volts (such as -5, +12, -12 volts) are installed in the memory chassis, special wiring within the memory chassis is required to route these voltages to the card slots. The I/O power cable (part number 53P0629) is routed to J30 on the I/O expansion chassis to provide power for peripheral controller cards. Without an expansion power supply in the system, a jumper plug (part number 53P0634) must be installed on J31 of the I/O expansion chassis.

The power cable connections for a system with a Varian 620 expansion power supply

SYSTEM INTERCONNECTION

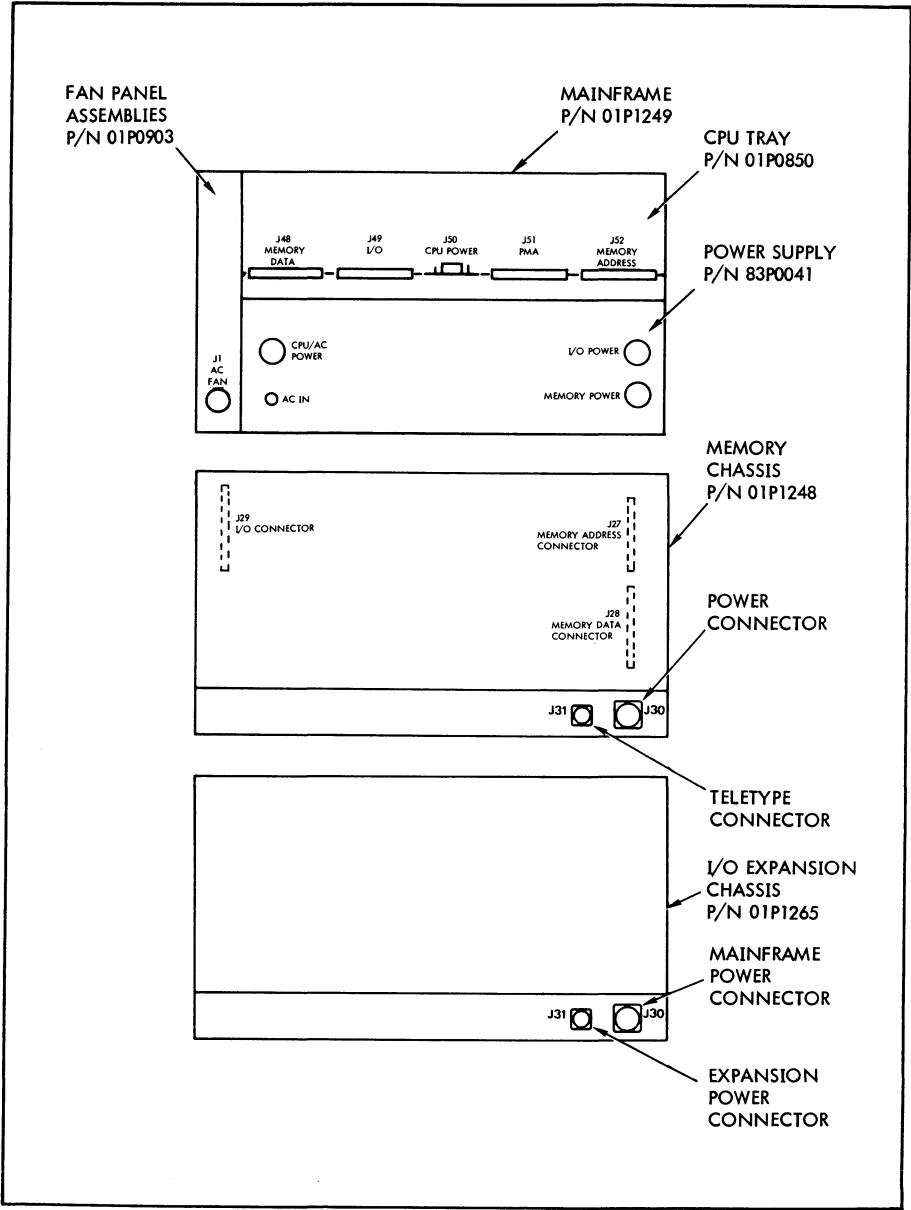
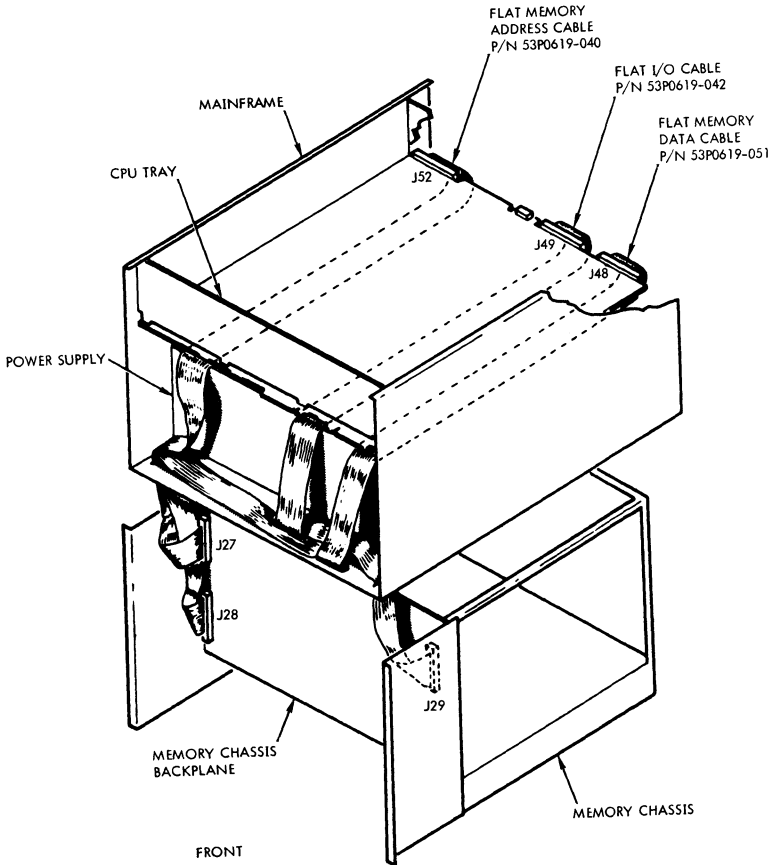
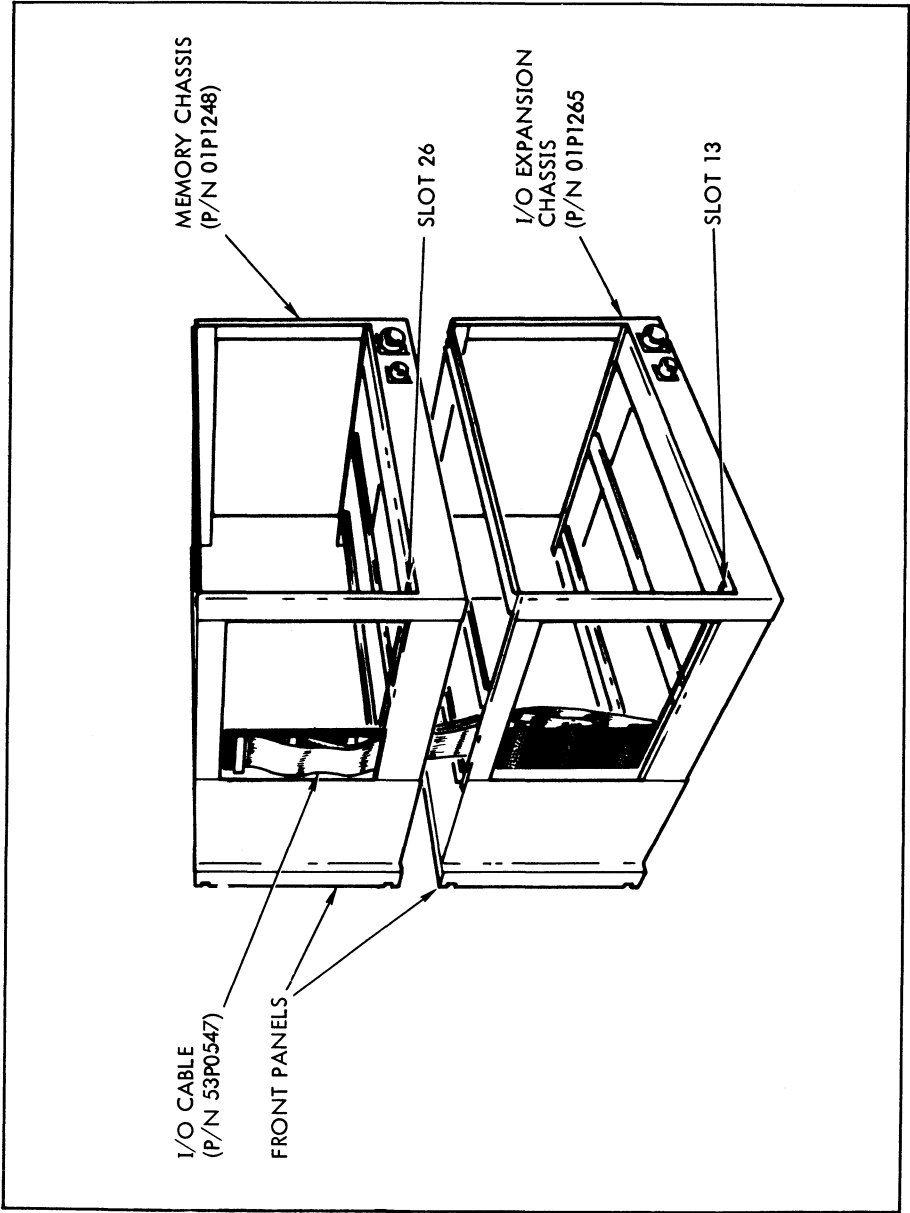


Figure 11-1. Chassis Connectors



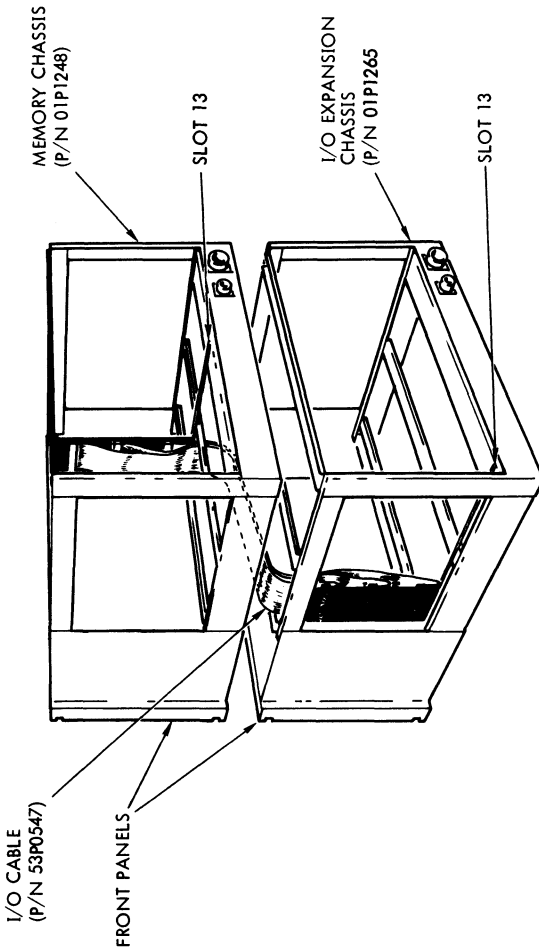
VT11-1333

Figure 11-2. Mainframe/Memory Chassis Interconnection



VTII-1331

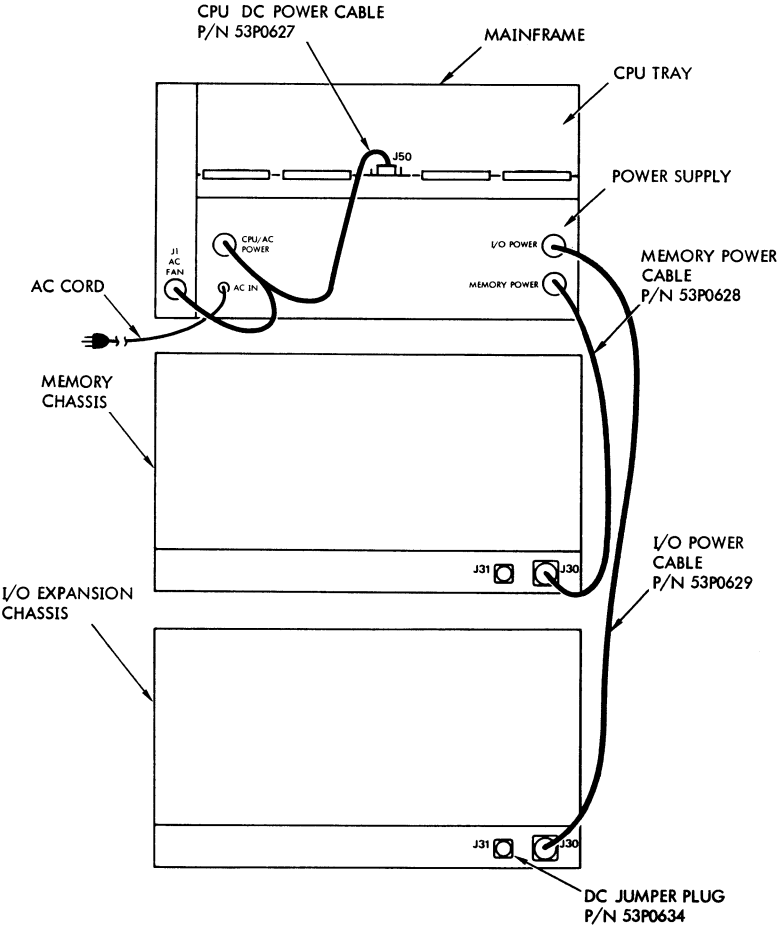
Figure 11-3. I/O Cable Installation in 32K Systems



NOTE:

AN I/O TERMINATION SHOE (P/N 44P0530) IS INSTALLED IN SLOT 2 OF RIGHT-HALF BACKPLANE IF I/O EXPANSION CHASSIS CONTAINS ONLY RIGHT-HALF BACKPLANE WIRING. IF THE I/O EXPANSION CHASSIS CONTAINS RIGHT-HALF AND LEFT-HALF BACKPLANE WIRING, THE TERMINATION SHOE IS INSTALLED IN SLOT 2 OF LEFT-HALF BACKPLANE.

Figure 11-4. I/O Cable Installation in 16K Systems



VTII-1334

Figure 11-5. Power Interconnection Without Expansion Supply

are illustrated in figure 11-6. The connections are the same as in figure 11-5 except a turn-on power cable connects to the expansion power supply at TB-1, points 5, and 6 and another cable from the supply to the I/O expansion chassis at T30.

Teletype Interconnection

A Teletype cable (part number 53P0148) connects to J31 on the memory chassis connector panel. The three Teletype signals (TTXX-T, TTSR-T, TTRX-T) are routed to the Teletype controller in the CPU tray via memory chassis connector J27, through the memory address cable (part number 53P0619-040), to connector J52 on the CPU tray.

System Interrupt Interconnection

System interrupt priority is established by the connecting of internal computer options in a priority chain. Options that can be included in the priority chain are: memory protection (MP), power failure/restart (PF/R), real-time clock (RTC), priority interrupt module (PIM), buffer interlace controller (BIC), and the INT (interrupt) switch on the control panel. Peripheral controllers do not generate interrupt requests; this function is performed with the PIM.

Normally, the priority assignments are wired at the factory before the equipment is delivered; however, a description is presented in this section for the user who wishes to augment or change his system. The interrupt priority assignment is unique for each computer system and specific information concerning each system is included as part of the system documentation delivered with the equipment.

Interconnection of options located in the same chassis is accomplished by connecting the Priority Out (PRNX - I) signal of

the first-priority option to the Priority in (PRMX - I) signal of the next-lower-priority option. This process continues until all subsequent priorities within the chassis are assigned. The priority-in signal of the first-priority option (except MP) is connected to ground. Because MP must always have first priority, it is supplied without a priority-in signal.

To establish a priority chain for options in separate chassis, four priority lines are available at pins 12, 14, 16, and 18 in the I/O cable (part number 53P0619-042, see table 11-3). The unused pins of the I/O expansion cable (part number 53P0547, see table 11-4) can be used as priority lines by wiring them into the system as required. The designation and cable pin assignments of four typical priority lines in this cable are listed as follows:

Signal	Pins
PR1X - I	37
PR2X - I	39
PR3X - I	41
PR4X - I	42

Figure 11-7 illustrates typical interrupt priority connections for a computer system with the MP, PF/R, and INT switch in the mainframe, and a BIC and PIM in the I/O expansion chassis. The connection between the first two priorities (MP and PF/R) is made on the bottom side of the CPU tray. The connection between the second and third priorities (PF/R and BIC) is routed out of the CPU tray via J49, and through the PR1X - I line of the flat I/O cable (part number 53P0619-042) to J29 on the memory chassis backplane. It is then wired to slot 26 of the memory chassis where it is routed through the PR1X - I line of the flat I/O expansion cable (part number 53P0547) to pin 37 of slot 13 of the I/O expansion chassis. Pin 37 of slot 13 is connected to pin 37 of the

SYSTEM INTERCONNECTION

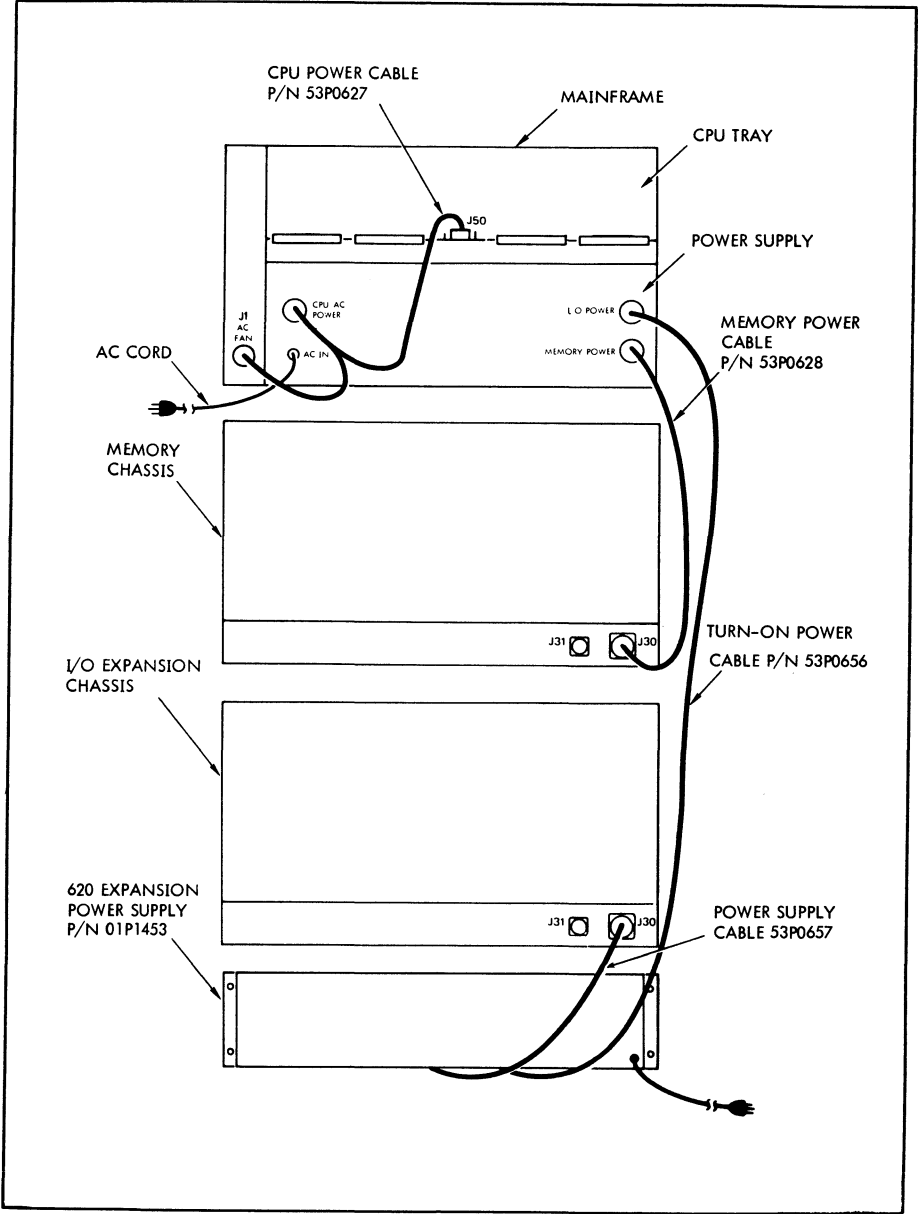
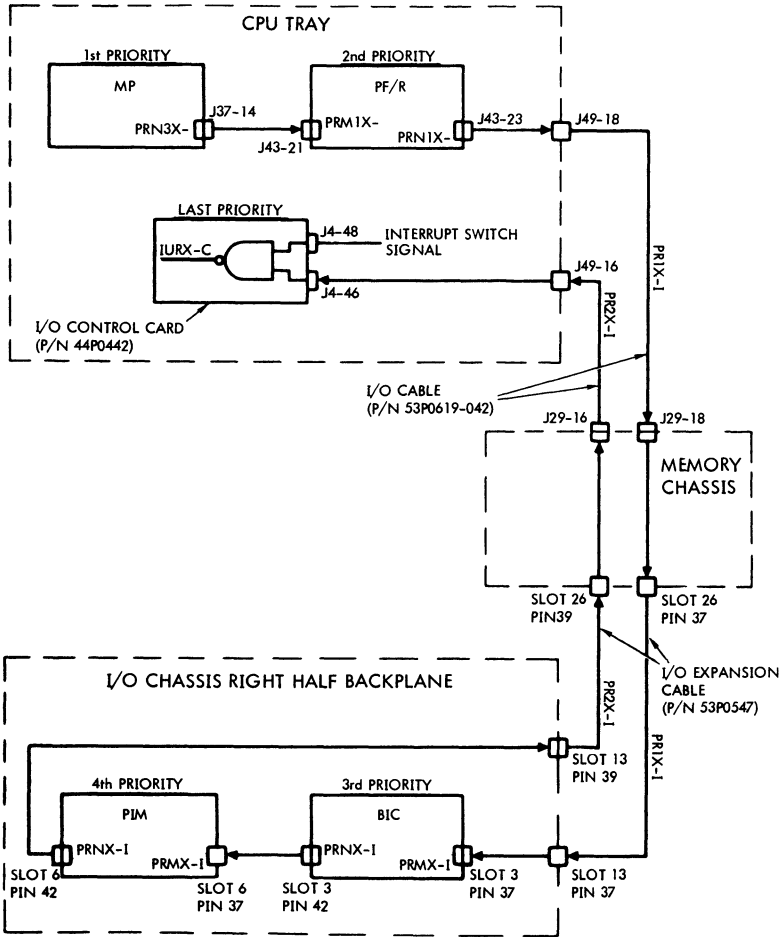


Figure 11-6. Power Interconnection With Expansion Supply



VTII-1337

Figure 11-7. Typical System Interrupt Priority Connections

SYSTEM INTERCONNECTION

BIC card (slot 3). The connection between the third and fourth priorities (BIC and PIM) is routed from pin 42 of slot 3 to pin 37 of slot 6 (PIM slot) on the I/O chassis backplane. The connection between the fourth and last priority (PIM and INT switch) is routed from pin 42 of the PIM card (slot 6) to pin 39 of slot 13, through the PR2X - I line of the I/O expansion cable (part number 53P0547) to slot 26 of the memory chassis, to J29 where it is routed through the PR2X - I line of the I/O cable (part number 53P0619-042) to pin 16 of J49 on the CPU tray. Pin 16 of J49 is connected to pin 46 of the I/O control card (part number 44P0442) where the interrupt signal (PRMX - I) is gated with the INT switch (INTSE +) signal to produce the interrupt request (IURX - C) signal.

PIM Interconnection

The optional priority interrupt module (PIM) has eight interrupt lines (IL00 through IL07) for connection with up to eight external interrupts. PIM interconnection consists of connecting these interrupt lines to selected peripheral controller cards, thus providing them with interrupt capabilities on a priority basis. For peripheral controllers that are located in the same computer chassis as the PIM, the interrupt lines are connected on the computer chassis backplane. For peripheral controllers located in a different chassis, the interrupt line connections are routed through I/O expansion cables.

BIC Interconnection

Control signal routing for the optional buffer interface controller (BIC) is divided into two parts. One part consists of the computer interface signals; the other, peripheral interface signals (B signals).

The computer interface signals are: FRYX - I, DRYX - I, IUAX - I, IUCX - I, SYRT - I, SERX - I, TPOX - I, and TPIX - I. The B signals are: DCEX - B, DESX - B, TAKX - B, CDCX - B, BCDX - B, TROX - B, and TRQX - B. The computer interface signals are routed to the mainframe via the I/O expansion cables. For a BIC connected to a peripheral controller (or controllers) in the same computer chassis, the B signals are routed via the chassis backplane; if the BIC is in another chassis, the B signals are routed via an I/O expansion cable. If a system contains more than one BIC, the seven B signals must connect only between the BIC and its associated controller; these signal lines must be broken between BICs.

PMA Interconnection

The optional priority memory access (PMA) allows up to four external devices direct access to the 620/f-100 memory. A PMA request from an external device prevents the CPU from using memory. A controller (designed by the user or Varian is required to interface the optional PMA circuits to each external device. The controllers can be installed in specially wired I/O card slots of a standard 620/f-100 expansion chassis, or in a chassis provided by the user. Signals from the optional PMA circuits in the mainframe are routed to the controller via a cable that mates with connector J51 at the rear of the mainframe (figure 11-1).

Cable Pin Assignments

Tables 11-1 through 11-8 list pin assignments for the following system interconnection cables:

SYSTEM INTERCONNECTION

- | | |
|---|---|
| a. Memory address cable (part number 53P0619-040) | e. CPU power cable (part number 53P0627) |
| b. Memory data cable (part number 53P0619-051) | f. Memory power cable (part number 53P0628) |
| c. I/O cable (part number 53P0619-042) | g. I/O power cable (part number 53P0629) |
| d. I/O expansion cable (part number 53P0547) | h. Teletype cable (part number 53P0148) |

Table 11-1 Memory Address Cable

Pin	Signal	Pin	Signal
1	GRD	26	ABS05 -
2	PFA	27	GRD
3	GRD	28	ABS06 -
4	BZY -	29	GRD
5	GRD	30	ABS07 -
6	MDR -	31	GRD
7	GRD	32	ABS08 -
8	RW -	33	GRD
9	GRD	34	ABS09 -
10	MST -	35	GRD
11	GRD	36	ABS10 -
12	RESET	37	GRD
13	GRD	38	ABS11 -
14	PFF	39	GRD
15	GRD	40	ABS12 -
16	ABS00 -	41	GRD
17	GRD	42	ABS13 -
18	ABS01 -	43	GRD
19	GRD	44	ABS14 -
20	ABS02 -	45	GRD
21	GRD	46	GRD
22	ABS03 -	47	TTX - T*
23	GRD	48	TTSR - T*
24	ABS04 -	49	TTRX - T*
25	GRD	50	GRD

* Teletype signals.

SYSTEM INTERCONNECTION

Table 11-2. Memory Data Cable

Pin	Signal	Pin	Signal
1	GRD	26	MD09 -
2	MD00 -	27	MB09 -
3	MB00 -	28	MD10 -
4	MD01 -	29	MB10 -
5	MB01 -	30	GRD
6	GRD	31	MB11 -
7	MB02 -	32	MD11 -
8	MD02 -	33	GRD
9	GRD	34	MD12 -
10	MD03 -	35	MB12 -
11	MB03 -	36	MD13 -
12	MD04 -	37	MB13 -
13	MB04 -	38	GRD
14	GRD	39	MB14 -
15	MB05 -	40	MD14 -
16	MD05 -	41	GRD
17	GRD	42	MD15 -
18	MD06 -	43	MB15 -
19	MB06 -	44	* MD16 -
20	MD07 -	45	* MB16 -
21	MB07 -	46	GRD
22	GRD	47	* MB17 -
23	MB08 -	48	* MD17 -
24	MD08 -	49	GRD
25	GRD	50	GRD

*Not used on 16-bit system

Table 11-3. I/O Cable

Pin	Signal	Pin	Signal
1	GRD	26	DRYX - I
2	IUJX - I	27	GRD
3	GRD	28	FRYX - I
4	IURX - I	29	GRD
5	GRD	30	ETR + *
6	IUCX - I	31	EB15 - I
7	GRD	32	EB14 - I
8	IUAX - I	33	EB13 - I
9	GRD	34	EB12 - I
10	SYRT - I	35	EB11 - I
11	GRD	36	EB10 - I
12	PR4X - I	37	EB09 - I
13	GRD	38	EB08 - I
14	PR3X - I	39	EB07 - I
15	GRD	40	EB06 - I
16	PR2X - I	41	EB05 - I
17	GRD	42	EB04 - I
18	PR1X - I	43	GRD
19	GRD	44	EB03 - I
20	TPOX - I	45	EB02 - I
21	GRD	46	EB01 - I
22	TPIX - I	47	+ 3V
23	GRD	48	EB00 - I
24	SERX - I	49	+ 3V
25	GRD	50	GRD

* External timing source for RTC.

SYSTEM INTERCONNECTION

Table 11-4. I/O Expansion cable

Pin	Signal	Pin	Signal
1	GRD	35	TPOX -I
2	EB00 -I	36	RT12 -
3	RT01 -	37	PR1X -I
4	EB01 -I	38	RT13 -
5	RT02 -	39	PR2X -I
6	EB02 -I	40	RT14 -
7	RT03 -	41	PR3X -I
8	EB03 -I	42	PR4X -I
9	RT04 -	43	SYRT -I
10	EB04 -I	44	IUAX -I
11	EB05 -I	45	IUCX -I
12	EB06 -I	46	IURX -I
13	EB07 -I	47	IUJX -I
14	EB08 -I	48	GRD
15	EB09 -I	49	TRQX -B
16	EB10 -I	50	TROX -B
17	EB11 -I	51	RT15 -
18	EB12 -I	52	BCDX -B
19	EB13 -I	53	RT16 -
20	EB14 -I	54	CDCX -B
21	EB15 -I	55	RT17 -
22	RT05 -	56	DCEX -B
23	*	57	RT18 -
24	RT06 -	58	TAKX -B
25	*	59	RT19 -
26	RT07 -	60	DESX -B
27	FRYX -I	61-73	
28	RT08 -	74	
29	DRYX -I	75-99	
30	RT09 -	100	GRD
31	SERX -I	101-115	
32	RT10 -	116	+ 3V dc
33	TPIX -I	117-121	
34	RT11 -	122	GRD

*Would be used for EB16 -I and EB17 -I in 18 bit systems

Table 11-5. CPU Power Cable

Signal	Power Supply	CPU Tray	Fan Panel
+3	TB3-6	P50-1	----
Return	TB3-5	P50-2	----
+5V	TB3-3	P50-3	----
Return	TB3-2	P50-4	----
+5V	TB3-1	P50-5	----
Return	TB3-2	P50-6	----
+5V	RB3-1	P50-7	----
Return	TB3-2	P50-8	----
+5V	TB3-3	P50-9	----
Return	TB3-2	P50-10	----
-5V	TB3-4	P50-11	----
Return	TB3-5	P50-12	----
115V fan	TB2-2	----	P1-A
115V return	TB2-4	----	P1-B
24V ac	S1-B	----	P1-C
Return	TB2-8	----	P1-D
Remote on/off	TB2-1	----	P1-E
Chassis GRD	E37	----	P1-F

Table 11-6. Memory Power Cable

P30 Pins	Signal
1	+3V
2	---
3	Chassis GRD
4	----
5	115V fan
6	115V return
7	---
8	---
9	---
10	PFA
11	PFA return
12	+12V
13	-12V
14	+5V
15	-5V
16	GRD
17	Return

Table 11-7. I/O Power Cable

P30 Pins	Signal
1	+3V
2 through 11	No connection
12	+12V
13	-12V
14	+5V
15	-5V
16	Return
17	Return

Table 11-8. Teletype Cable

P31 Pins	Signal
1	TTSR - T
2	TTRX - T
3	TTXX - T

SECTION 12 - SYSTEM OPERATION

The Varian 620/f-100 operator's console (i.e., the computer's control panel and a Teletype) provides all the controls, displays, and printed output needed for operating the computer system.

Program Execution

The Varian 620/f-100 requires very little preparation before a program can be executed. Assuming that the system, including peripherals, is properly installed and connected to an ac power source, the following procedure is followed to make a cold start (i.e., when a new system is being initialized or the contents of memory are unknown).

- a. Turn on computer power by placing the power keyswitch on the control panel (figure 12-1) to PWR ON.
- b. Initialize the system by pressing RESET. Clear all registers by momentarily pressing each register selection switch, pressing RESET each time.
- c. Load the appropriate bootstrap loader routine either manually or automatically (if the system contains the ABL option). Refer to the Switches and Indicators section for the BOOTSTRAP switch description, and the Manual Operations section for manual loading of the bootstrap loader.
- d. Load the binary load/dump program (BLD II, section 18) using the Teletype or high-speed paper tape reader (depending on the bootstrap loader selected). Verify after loading that

the P register contains the proper starting address (section 18).

- e. Load the object program using the same paper tape reader as that used for BLD II.
- f. Press the RUN and START switches on the control panel.

This section describes control panel switches and indicators, and implementation of the above procedure and manual operations.

Switches and Indicators

The control panel of the Varian 620/f-100 is illustrated in figure 12-1.

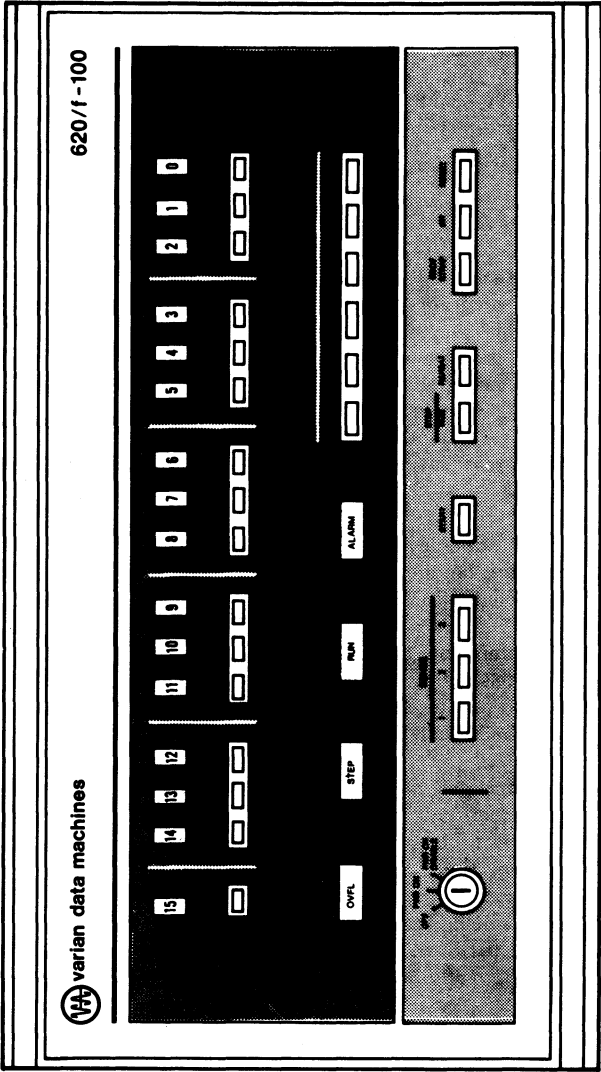
Power Switch

The key-operated power switch controls the ac input to the 620/f-100 power supply.

In the PWR OFF position, ac input to the power supply primary is disabled.

In the PWR ON position, there is ac power to the power supply primary and the system should be fully operational.

In the PWR ON DISABLE position, there is ac power to the power supply primary and the computer is operational. However, all control switches are disabled except the power switch itself. The settings of SENSE switches at the time of disabling is stored in the CPU and remain until the panel is enabled and the switch settings are changed. Pressing any other switch while



VTII-1446

Figure 12-1. Varian 620/f-100 Control Panel

the power switch is in PWR ON DISABLE has no effect.

The control panel and power supply indicator lights are functional when the power switch is in PWR ON or PWR ON DISABLE.

The key can be removed from the power switch in any of the three positions. To turn off the computer, place the power switch in the PWR ON position, lift the STEP/RUN switch, then turn the power switch to PWR OFF.

STEP/RUN Switch and STEP and RUN

Indicators

When the STEP/RUN switch is up, the 620/f-100 is in step mode and the step indicator is lit. When the switch is down, the computer is in run mode. The RUN indicator lights after START is depressed.

If the computer is in step mode:

- a. Pressing the STEP/RUN switch to RUN position primes the computer to enter the run mode when the START switch is pressed.
- b. Pressing the START switch executes the instruction in the I register, and fetches the next instruction from the address specified by the contents of the P register and places it in the I register.

If the computer is in run mode:

- a. Lifting the STEP/RUN switch to STEP position halts the 620/f-100 after completing the execution of the current instruction and fetches the next instruction and sets it in the I register. The RUN indicator goes out and the STEP indicator lights.

- b. Pressing the START switch starts the program at the address specified by the P register after executing the instruction in the I register.

Place the STEP/RUN switch in the RUN position for loading the automatic bootstrap and in the STEP position for manual loading of the bootstrap and for manual data entry or register display.

BOOTSTRAP Switch

BOOTSTRAP is a momentary, spring-loaded switch that is functional in 620/f-100 systems containing the ABL option. In other 620/f-100 systems this switch is present on the control panel, but it is not connected.

The bootstrap program permits the loading of the binary load/dump program into memory. Before the automatic bootstrap is loaded into memory, the binary load/dump tape should be inserted into the paper tape reader with the first binary frame at the reading station.

To load the automatic bootstrap program:

- a. Set the power switch to PWR ON.
- b. Set the STEP/RUN switch to RUN; this loads the bootstrap program into memory.
- c. Press and release BOOTSTRAP.

If the system does not contain the ABL option, load the bootstrap program manually (table 12-1).

START Switch

START is a momentary, spring-loaded switch. Pressing it with the STEP/RUN

SYSTEM OPERATION

switch in the RUN position places the computer in the run mode and starts the program. Pressing the START switch when the computer is in the step mode executes the instruction in the I register (except HLT), and fetches the next instruction from the address specified by the contents of the P register and places it in the I register.

REGISTER Switches

Pressing one of the five REGISTER switches selects the designated register (X, B, A, I, or P) for display or entry.

Only one register can be selected at a time. Pressing two or more REGISTER switches simultaneously disables the selection logic and ORs the front panel register display. The I register will display the I register regardless of other selections.

Register Entry Switches and Display Indicators

The 16 indicators across the top of the 620/f-100 control panel display the contents of a selected register. Data are entered into registers on the corresponding register entry switches under the indicators. The indicators and switches are read from left to right, bits 15 to 0. An illuminated indicator shows that that bit contains a one. For negative data, the sign bit (bit 15) is a one. The indicators and switches are divided into groups of three for ease in reading octal configurations.

To display the contents of a register, switch the STEP/RUN switch to STEP and press the REGISTER switch for the desired register.

The display indicators light when they correspond to register bits that contain

ones. To remove the display, pull up on the REGISTER switch and the indicators go out.

To enter data or instructions in a register:

- Display the contents of the register.
- Enter ones by pressing down on the register entry switches corresponding to the bits to be set.
- Enter zeros in the other bits by pulling up on all other register entry switches. The indicator lights do not change when the register entry switches are manipulated. They still display the contents of the register.
- When the desired configuration is entered on the register entry switches, press LOAD. This loads the register with the configuration entered on the switches, and the indicators change to display this new configuration in the register.

To enter data into core memory:

- Load into the I register a storage instruction (STA, etc).
- Select the register specified by the storage instruction in step a.
- Load the selected register using the register entry switches.
- Press START to execute the instruction in the I register. This stores the contents of the specified register at the effective memory address.

The TSA instruction can also transfer data entered on the control panel switches to the A register.

LOAD Switch

LOAD is a momentary, spring-loaded switch. When the 620/f-100 is in step mode and a register has been selected, pressing this switch loads the register with the bit configuration entered on the register entry switches.

REPEAT Switch

REPEAT is a toggle switch that is operative in both step and run modes. To repeat an instruction contained in the I register, press REPEAT, and then press START. The instruction is executed again and the program counter advances. However, the contents of the I register remain unchanged.

To run a program, REPEAT must be off.

SENSE Switches

The three SENSE switches are toggle switches permitting program modification by the operator. When the program contains instructions dependent on the setting of these switches, jumps and executions occur only when the switch condition is met.

To set a SENSE switch, press down. To reset it, lift. Operations dependent on the position of this switch are executed if the switch is in the position indicated by the instruction.

EXAMPLE

A program can be written so that the operator can obtain a partial total of a column of figures being added by use of the JSS1 (jump if SENSE switch 1 is set) instruction. The program writes individual entries

as long as SENSE switch 1 is not set. When the operator wants a partial total, he sets the switch. The program then jumps to an instruction sequence that prints the desired information.

INT (Interrupt) Switch

INT is a momentary, spring-loaded switch used to interrupt the 620/f-100 computer. It is functional only when the 620/f-100 is in the run mode.

Pressing INT interrupts to memory address 000000.

RESET Switch

RESET is a momentary, spring-loaded switch used for initialization control and for stopping I/O operations. Pressing this switch halts the 620/f-100 and initializes the computer and peripherals. This switch is electrically interlocked with the STEP/RUN switch and is disabled when the latter is in RUN.

Note that this switch is not a display reset.

OVFL (Overflow) Indicator

OVFL lights whenever an overflow condition exists.

ALARM Indicator

ALARM lights to signal an overheated system. If the power switch key is accessible, turn the power switch to PWR OFF and call the Varian Customer Service engineer.

If the power switch key is not accessible, turn off the power switch located on the back of the power supply, or pull the main

SYSTEM OPERATION

plug, and call the Varian Customer Service engineer.

Manual Operations

With the 620/f-100 in step mode, data or instructions can be manually transferred to or from memory or stored programs can be manually executed.

Note that the I register contains the instruction being executed, while the P register points to the address of the following instruction.

Loading the Bootstrap Loader Manually

The instruction codes of the bootstrap loader programs are listed in table 12-1. After the computer power is on and the system initialized:

- a. Load instruction 054000 in the I register (store A relative to P).
- b. Place REPEAT in the down position and set STEP/RUN to STEP.
- c. Load the starting memory address (007756) of the bootstrap loader into the P register.
- d. Load the appropriate bootstrap instruction into the A register, pressing START after loading each instruction. If the high-speed paper tape reader is to be used for subsequent program input, select the column headed High-Speed Reader Code in table 12-1; if using the Teletype paper tape reader, select the column headed Teletype Reader Code. The computer stores the A register contents into the address specified by the P register, which is incremented by one after each instruction is loaded.

To determine that the bootstrap loader has been correctly loaded:

- a. Initialize the CPU by pressing RESET.
- b. Clear all registers by momentarily pressing each register switch, pressing RESET each time.
- c. Load instruction 014000 (load A relative to P) in the I register.
- d. Load the starting memory address (007756) in the P register. Keep REPEAT in the down position.
- e. Select the A register and press STEP. The contents of each memory address are sequentially displayed each time STEP is pressed.
- f. If an error is found, correctly reload the erroneous instruction codes into memory.

NOTE

The P register error address is always the error address plus one.

The Varian 620/f-100 is now ready for use. The procedures for loading the binary load/dump program and subsequent object programs are given in section 18.

Loading, Displaying, and Altering

Memory

To load data or instructions into memory, to display the contents of memory, or to alter contents of memory, follow the procedures given under the description of the register entry switches and display indicators above.

Table 12-1. Bootstrap Loader Routines

Address	High-Speed Reader Code	Teletype Reader Code	Symbolic Coding
007756	102637	102601	READ CIB RDR
007757	004011	004011	ASLB NBIT - 7
007760	004041	004041	LRLB 1
007761	004446	004446	LLRL 6
007762	001020	001020	JBZ SEL
007763	007772	007772	(Memory address)
007764	055000	055000	STA 0,1
007765	001010	001010	JAZ LHLT + 1
007766	007000*	007000*	(Memory address)
007767	005144	005144	IXR
007770	005101	005101	ENTR INCR 1
007771	100537	102601	SEL SEL RDON
007772	101537	101201	SEN IBFR,READ
007773	007756	007756	(Memory address)
007774	001000	001000	JMP * - 2
007775	007772	007772	Memory address)

NOTE

The bootstrap loader routine is always loaded into the highest address of the first 4K memory increment, regardless of available memory. BLD II relocation and adaptation to the specified input device are described in section 18.

* Replace this code with 007600 if the test executive of MAINTAIN II (refer to document number 98 A 9952 060) is to be loaded and executed.

SYSTEM OPERATION

Loading Sequential Memory Addresses

To load a sequential group of memory addresses:

- a. Set STEP/RUN to STEP and press REPEAT.
- b. Load the P register with the base address.
- c. Load into the I register a storage instruction (STA, etc.) with 100 in the M field (relative addressing), and zero in the A field.
- d. Select the register specified by the storage instruction in step c.
- e. Load the selected register using the data entry switches.
- f. Press START to execute the instruction in the I register.
- g. Repeat steps e and f until all instructions are loaded. The next address to be loaded can be observed by displaying the P register.

Displaying Sequential Memory Addresses

To display the contents of a group of sequential memory addresses:

- a. Place STEP/RUN in STEP, and press REPEAT.
- b. Load the P register with the base address.
- c. Load into the I register a loading instruction (LDA, etc.) with 100 in

the M field (relative addressing), and zero in the A field.

- d. Select the register specified by the loading instruction in step c.
- e. Press START once for each memory location to be displayed.

Executing of a Stored Program

To execute a stored program manually:

- a. Select step mode and turn off REPEAT.
- b. Set the P register to the first address of the program.
- c. Clear the I register.
- d. Press START.
- e. Press START again to execute the instruction and to load the next instruction into the I register.
- f. Repeat step e once for each instruction.

Repeating an Instruction

To repeat an instruction manually:

- a. Press the REPEAT switch down.
- b. Press START. This advances the P register but inhibits loading the I register. Thus, pressing START again executes the same instruction.

SECTION 13 - MAINTENANCE

Integrated-circuit (IC) design reduces the occurrence of malfunctions in Varian 620-100 systems. Convenient packaging methods make all system elements accessible for troubleshooting and maintenance in the rare case when the system malfunctions.

Complete troubleshooting and maintenance information for the Varian 620-100 systems is presented in the Varian 620-100 maintenance manuals, Volumes 1 and 2 (document numbers 98 A 9905 150 and 98 A 9908 150), in the Varian 620 Test Programs Manual (document number 98 A 9952 060), in applicable computer and I/O option and peripheral controller manuals, and in manufacturer's instruction manuals with peripheral devices.

This section of the Handbook outlines, in general terms, routine maintenance and troubleshooting concepts.

Routine Maintenance

The PF/R (sections 4 and 32) is an attractive addition to the Varian 620-100 system. It provides an orderly shutdown in case of power failure or turn-off, and automatically restarts the interrupted program when power is restored.

To prevent accidental setting of control panel switches during computer operation, turn the power switch to the DISABLE position.

The Varian 620-100 system requires little routine maintenance other than a periodic check of memory timing. A complete set of waveforms is given in the maintenance manual.

Cooling fans for the memory and power supplies, and the mechanical equipment interfaces manufactured by Varian Data Machines, are permanently lubricated and require no routine attention.

To ensure effective maintenance of the Varian 620-100:

- a. Study the documentation furnished with the equipment.
- b. Assess system performance with adequate test equipment.
- c. Use the available maintenance aids and test programs.
- d. Apply orderly and logical troubleshooting techniques.

Test Equipment

Table 13-1 lists the test equipment recommended for computer maintenance. These items also satisfy maintenance requirements of the peripheral controllers.

Test Programs

The MAINTAIN II test program system briefly outlined in section 22, and described in detail in the Varian 620 Test Programs Manual (document number 98 A 9952 060), verifies correct system operation. This system tests all phases of system operation, including memory, machine instructions, computer and I/O options, and peripherals and their controllers.

MAINTAIN II aids in preventive maintenance by determining whether the system

Table 13-1. Recommended Test Equipment

Item	Description
Oscilloscope	Tektronix, type 547 (or equivalent) with dual-trace plug-in unit
Multimeter	Simpson 260 (or equivalent)
Card Extender	Varian part numbers DM312 and DM265 extender cards
Card Puller	Titchener 1731 (or equivalent)
Wire-wrap Gun	Gardener-Denver 14R2 (or equivalent)
Soldering Iron	3-watt pencil type
Wire Stripper	Thermo-strip type
Assorted Hand Tools	Screwdrivers, spin-tight wrenches, long-nosed pliers, etc.

is actually malfunctioning, and, in most cases, helps to isolate the error if one exists. When the system is definitely malfunctioning and the exact nature of the trouble is not known, test programs that exercise the suspected area of the fault can be loaded and executed independent of other MAINTAIN II programs.

Circuit Card Accessibility

Circuit cards comprising the Varian 620-100 systems can be installed on the DM312 and DM265 extender card (table 13-1). This configuration allows the circuit card to extend so that signals can then be monitored with an oscilloscope at the IC terminals.

Troubleshooting

Although many troubleshooting techniques can be applied, there is no substitute for an ordered, logical analysis of the problem and isolation of the cause of a failure to the level of the faulty component. Such an analysis can be based only on thorough knowledge of the equipment design.

One of the most valuable troubleshooting aids available to the technician is the computer's control panel. Using the switches and indicators, the operator can execute simple procedures manually to check out computer operations. The maintenance manuals list these procedures.

In general, the characteristics of a failure can indicate the faulty section of the system; for example:

- a. A CPU failure usually produces errors in a large percentage of the MAINTAIN II test programs. If the failure is catastrophic and all instructions fail, the cause is usually in the timing, decoding, and control sections. Incorrect arithmetic operations or incorrect incrementation of the P register indicates that the arithmetic/logic and register sections are at fault.
- b. Memory failures are indicated by repeated, unprogrammed halts in the execution of a program, or by completely random instruction sequencing. Malfunctions of a single memory bit or word are rare and usually require special test procedures (refer to the appropriate Varian 620-100 maintenance manual).
- c. Failures in I/O operations are associated only with the logic of the failing device. Such failures can be easily diagnosed if malfunctions occur only during the execution of I/O routines.

General troubleshooting steps are summarized below.

Define the problem thoroughly. For example, if the ADD instruction (section 16) does not produce correct results, check that the fault is a function of the sign (plus

or minus), of the carries, or of some other element. Verify that the operation registers are being loaded properly. Use the displays, connector pins, and IC terminals for gathering the necessary data.

Look for obvious solutions. Make sure that a malfunction has actually occurred. Relate problems to recent events, such as cleaning and servicing. Look for improperly set controls or test equipment and accidental disconnections of plugs, etc. Consider miscellaneous temporary failure, such as mechanical jamming of peripheral equipment.

Isolate the fault to a functional area, such as memory control, arithmetic/logic, operation register, I/O, peripheral controller, or peripheral device. This is generally a straightforward process of eliminating areas that are operating properly.

Analyze the faulty area. Use the logic and timing diagrams, and observe waveforms to isolate the problem to an individual replaceable element. Make sure that the computer is in step mode (STEP indicator on) before removing input power.

Correct the fault by replacing the faulty circuit card or component. Before restoring power, take any necessary measures to prevent recurrence of the failure.

Restore the system to normal operation. Verify proper operation by running the test programs.

SECTION 14 - COMPUTER WORD FORMATS

A word in the Varian 620-100 computers is 16 bits long and can contain data, a memory address, or an instruction. The words are normally expressed in the octal numbering system.

The octal system of assigning numerical values to binary forms deals with groups of three binary positions; each group is considered a single digit. Thus, in any octal digit, there are eight possible binary configurations:

Binary	Octal
000 =	0
001 =	1
010 =	2
011 =	3
100 =	4
101 =	5
110 =	6
111 =	7

EXAMPLE:

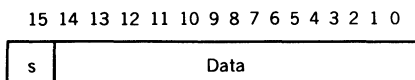
Bit Position

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	1	0	0	0	0	1	1	0	1	0	1	1
0	2			4			1	5			3				

Numbering systems are discussed in detail in appendix B. Octal-to-decimal conversion tables are given in appendixes D and E.

Data Word

Data words contain operands, i.e., data upon which an operation is performed. These words have the format:

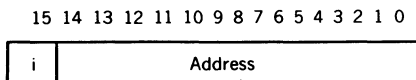


The most significant bit (bit 15) is the sign bit. It is one for negative numbers and zero for positive numbers. The other 15 bits (0-14) contain the data itself.

Negative numbers are represented in two's complement form. Zero is considered positive.

Address Word

Address words have the format:



In address words, the most significant bit is the indirect addressing bit. It is one for an indirect address and zero for a direct address. The other 15 bits contain the address.

Indirect addressing can extend to many levels. Each level adds one machine cycle

COMPUTER WORD FORMATS

(1.8 microseconds) to the basic execution time of an instruction.

Addressing is discussed in section 15.

Instruction Word

The Varian 620-100 series computers execute one- and two-word addressing and nonaddressing instructions. Figure 14-1 is a simplified flowchart of instruction-processing operations. Addressing mode information is given in section 15. Individual instructions are described in section 16.

Every instruction has an operation code defining the type of instruction, e.g.:

High-Order Bits 9-15			Instruction Type
0	0 0 0	0 0 1	Jump
0	0 0 0	0 1 0	Jump and mark
0	0 0 0	0 1 1	Execution
0	0 0 0	1 0 0	Shift/rotation
0	0 0 0	1 0 1	Register transfer/ modification
0	0 0 0	1 1 0	Extended and immediate addressing
0	0 0 0	1 1 1	Control
x	x x x	x x x	Load/store, arithmetic, and logic

where x = an operation code that differs for each instruction.

All instructions, except those of the one-word nonaddressing type, have, in addition, a mode (M) field and an address (A) field.

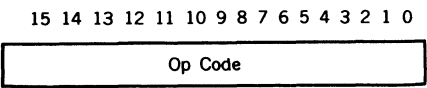
The M field specifies the addressing mode or the switch or register upon which the execution of the instruction depends.

The A field contains addressing information (section 15). In two-word instructions, the A field is contained in the second word.

- a. In instructions for loading, storing, arithmetic, and control operations, the A field contains an address, except in the halt instruction, where the value of the A field identifies the particular halt in a program.
- b. In immediate instructions, the address or operand is always the second word.
- c. In all other two-word formats, the second word contains the address. If bit 15 is one, the address is indirect. If bit 15 is zero, the address is direct.

One-Word Nonaddressing Instructions

This format consists of a 16-bit operation code standing alone:



It is used in control instructions ROF and SOF, which have configuration 0 000 111 in bits 9-15.

It is also used for instructions that shift or rotate the contents of registers. In shifts, the bits shifted out of the register are lost; but, in rotations, they are reloaded one at a time into the opposite end of the register. Shift/rotation instructions have

NOTE:
NOT APPLICABLE TO
SPECIAL INSTRUCTIONS
*SRE, LJMP, *JSR OR *BT

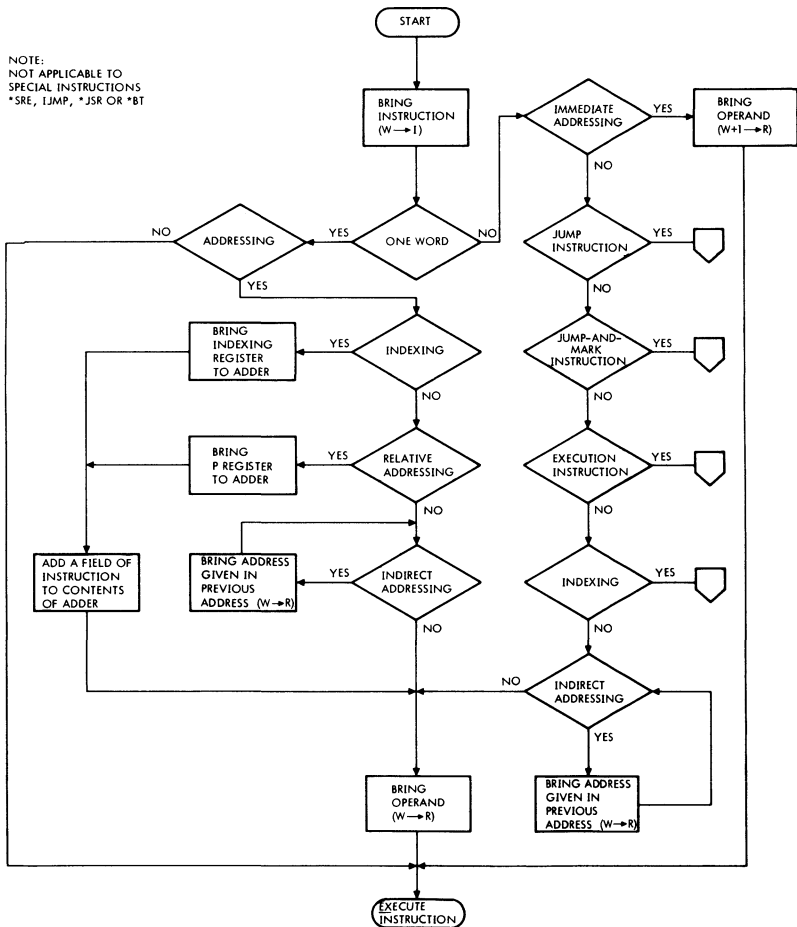


Figure 14-1. Instruction Processing, Simplified Flow

COMPUTER WORD FORMATS

the common and specific configuration 0 000 100 in bits 9-15.

One-Word Addressing Instructions

The first format of this type of instruction has a four-bit operation code followed by a three-bit M field and a nine-bit A field, with the A field being used for addressing. It is used for loading, storage, and arithmetic instructions:

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Op Code	M	A
---------	---	---

The operation code varies for each instruction. The A field contains an address in the addressing mode specified by the M field (section 15, table 15-1).

The halt instruction is an exception in that the A field does not contain an address. It can contain a value that identifies the specific halt in the program.

The second one-word addressing instruction type is actually **pseudoaddressing** in that registers, rather than memory, are addressed. These instructions have a ten-bit operation code followed by a three-bit M field and a three-bit A field, with both fields being used for pseudoaddressing of registers from which information is to be fetched and into which it is to be loaded.

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Op Code	M	A
---------	---	---

This format is used to transfer or modify register contents. Bits 9-15 have configuration 0 000 101. Bits 6 and 7 specify that

data be transferred from one register to another as follows:

w00	Unmodified	w10	Complemented
w01	Incremented	w11	Decrement

Bit 8 specifies conditional execution of the instruction depending upon the overflow indicator:

0cc	Execute instruction unconditionally
1cc	Execute only if the overflow indicator is set

The M field specifies the location from which the data to be transferred or modified are obtained (source), as follows:

000	Clears destination
001	A register
010	B register
100	X register

The A field specifies the location into which the modified and/or transferred data are to be placed, using the same values as the M field, except that 000 generates a no operation.

Two-Word Nonaddressing Instructions

This format comprises a 13-bit operation code, a three-bit M field in the first word, and a 16-bit operand in the second word:

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Op Code	M
Operand	

It is used for immediate-addressing instructions; thus, the address of the operand to be manipulated is implied in the second word.

Bits 6-15 of the operation code have configuration 0 000 110 00x. Bits 3-6 specify the manipulation; the configuration of these bits is the same as that of the operation code for the corresponding one-word addressing instruction.

The M field is 000, which precludes relative or indirect addressing or indexing.

The second word of the instruction is always the data to be processed. Thus, in the Load A Register Immediate (LDAl) instruction, the contents of the second word are loaded into the A register.

Two-Word Addressing Instructions

The two-word addressing instructions comprise the following categories:

- a. Extended-addressing versions of load/store, arithmetic, and logic instructions
- b. Jump and jump-and-mark instructions
- c. Execution instructions

Word two consists of an address composed of a direct/indirect addressing bit (bit 15) and a 15-bit address. If bit 15 is one, the address in the second word is indirect; if bit 15 is zero, the address is direct.

Extended-Addressing Instructions

This format comprises a 13-bit operation code, a three-bit M field in the first word, and an effective address in the second:

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Op Code										M	
i	Extended Address										

The highest-order bits of the operation code have configuration 0 000 110 00x. Bits 3-6 specify the manipulation; the configuration of these bits is the same as that of the operation code for the corresponding one-word addressing instruction.

The M field specifies the addressing mode (section 15, table 15-1).

The effective address in the second word is the extended address of the data upon which this instruction operates.

Jump and Jump-and-Mark Instructions

This format comprises a seven-bit operation code, a nine-bit M field in the first word, and an effective jump address in the second:

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Op Code										Function									
i		Jump Address																	

The operation code configuration is 0 000 001 for jump, and 0 000 010 for jump-and-mark, instructions.

For an unconditional jump or jump-and-mark instruction, all bits 8-0 are zeros. For other instructions in this group, each switch, register, or indicator to be examined is represented by a specific bit. The significance of these bits is described in section 15.

In jump instructions, the effective address in the second word is the address of the next instruction to be executed if the jump condition is met. The program then continues to execute instructions in the locations following the jump address.

COMPUTER WORD FORMATS

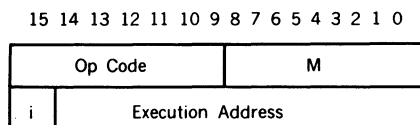
In jump-and-mark instructions, the effective address in the second word is the address where the contents of the P register are stored if the jump-and-mark condition is met. The next instruction to be executed is located in jump address plus 1. The program then continues to execute instructions following the one in jump address plus 1.

If the jump or jump-and-mark condition is not met, the program executes the next sequential instruction.

Figures 14-2 and 14-3 are flowcharts for jump and jump-and-mark instructions, respectively.

Execution Instructions

This format comprises a seven-bit operation code, a nine-bit M field in the first word, and an effective execution address in the second:



The operation code configuration is 0 000 011 for execution instructions.

The M field bits have the same significance as for the jump and jump-and-mark instructions.

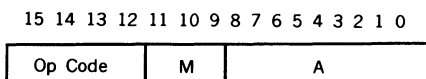
The effective address in the second word is the address of the next instruction to be executed if the execution condition is met. After executing that instruction, the program returns to the main program sequence and executes the next sequential instruction. The instruction at the effective address must be a one-word instruction.

Figure 14-4 is a flowchart for execution instructions.

I/O Instructions

The formats of two-word I/O instructions are identical with those of analogous internal instructions, except that bits 0-5 of the first word contain a device address. Bits 6 through 8 specify the device line involved for sense instructions.

The format of one-word I/O instructions is:



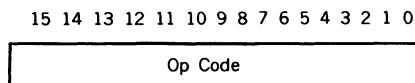
where the operation code always starts with a one; the M field denotes the register or function involved; and the device address has the value of the desired peripheral device address (section 7).

Instruction Format Summary

Instruction word formats are summarized below. Individual instructions are described in section 16.

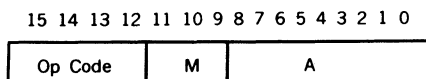
One-Word Nonaddressing Instructions

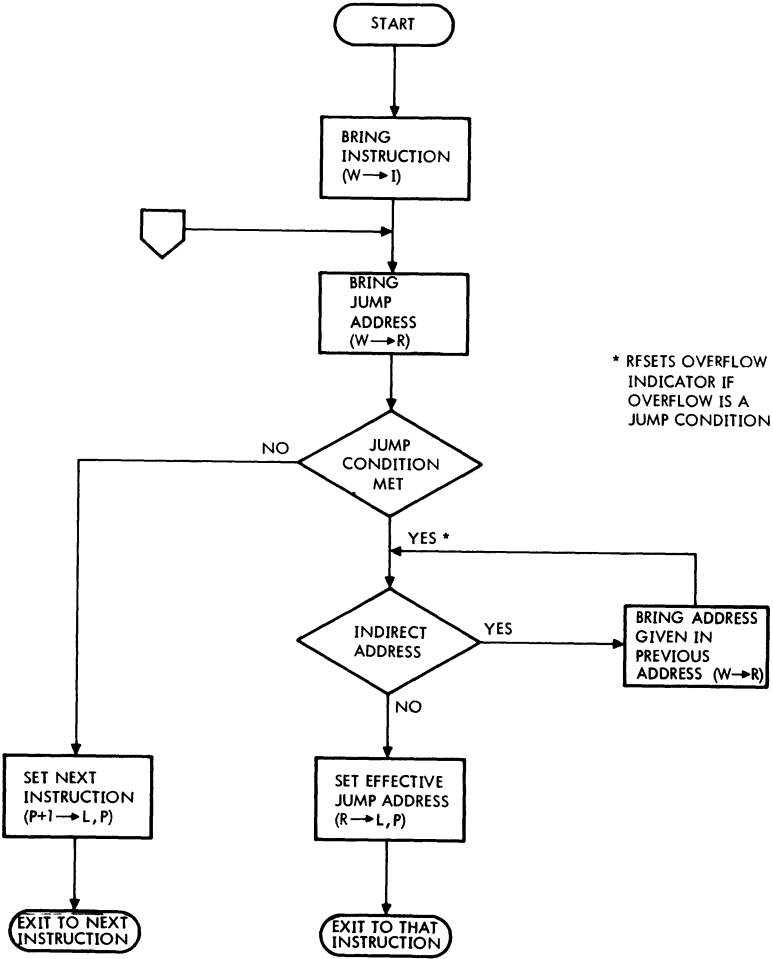
Shift/Rotation and Control Instructions



One-Word Addressing Instructions

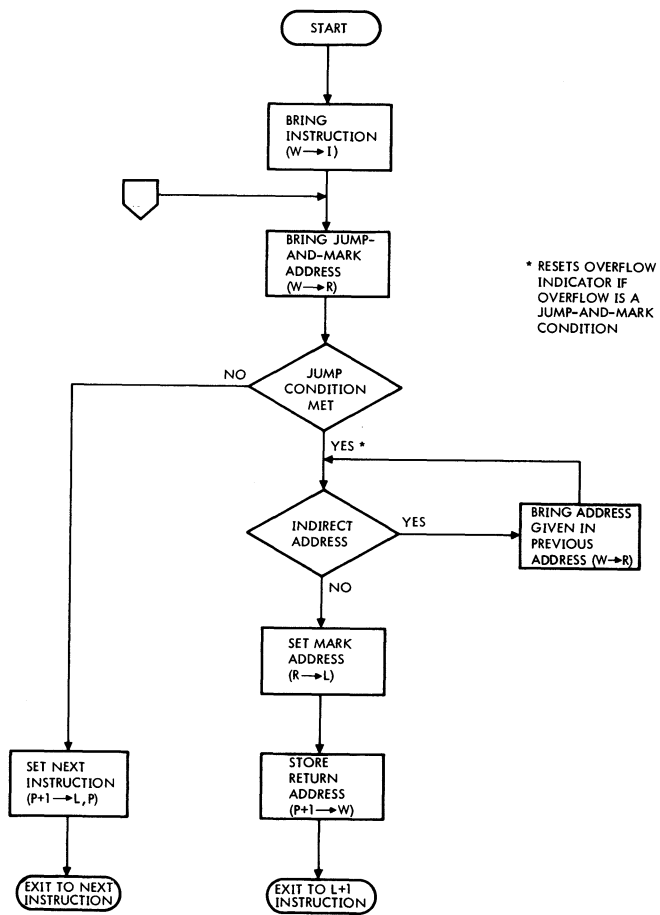
Load/Store, Arithmetic, and Logic Instructions





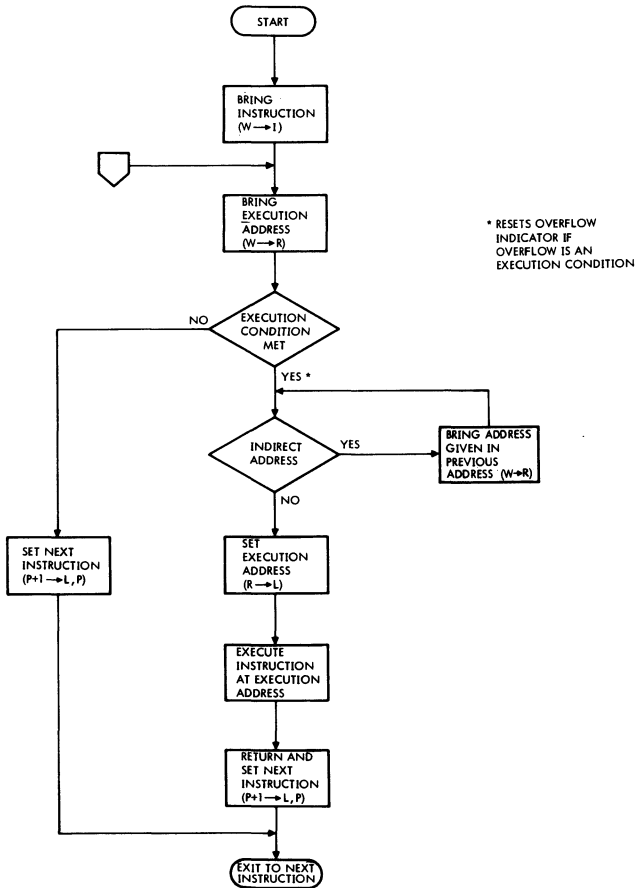
VT11-0681

Figure 14-2. Jump Instruction, General Flow



VT13-0245

Figure 14-3. Jump-and-Mark Instruction, General Flow



VT13-0246

Figure 14-4. Execution Instruction, General Flow

SECTION 15 - ADDRESSING MODES

The Varian 620-100 series computers feature the following addressing modes for increasing program efficiency:

- Direct
- Indirect
- Relative
- Indexed with the X or B register
- Extended
- Immediate

The instruction addressing mode is specified by the value of the M field as indicated in table 15-1. Additional information on instruction formats is given in section 14. Individual instructions are discussed in section 16.

Direct Addressing

In direct addressing, the address of the operand (i.e., the effective memory address) is contained within the instruction.

Direct-addressing one-word instructions have zero in the most significant bit of the M field (bit 11). The other two bits of the M field (bits 9 and 10) combine with the address (A) field (bits 0-8) to give an 11-bit address in the first 2,048 words of memory.

Any two-word addressing instruction can use direct addressing. The second word of

the instruction is the A field. A direct address is specified when bit 15 of the second word is zero.

Indirect Addressing

In indirect addressing, the A field contains an indirect address word in the first 512 words of memory. The contents of this address word can be either direct (bit 15 = 0), or indirect (bit 15 = 1) to implement multilevel indirect addressing.

Multilevel indirect addressing proceeds until an address in which bit 15 is zero is found. The contents of this address give the effective memory address. Each level of indirect addressing adds one cycle to the execution time of an instruction. In the 620/L-100, indirect addressing can be extended to any level; in the 620/f-100, to five levels with one-word instructions and to four levels with two-word instructions.

In indirect-addressing instructions, the M field is 111 (07).

Any two-word addressing instruction can use indirect addressing. The second word of the instruction is the A field.

Relative Addressing

Relative addressing is specified by an M field of 100 (04). In this addressing mode, the contents of the A field of an instruction are added to the contents of the P register plus one to form the effective address. This permits addressing locations

Table 15-1. Instruction Addressing-Mode Specifications

Addressing Mode	M Field	Effective Memory Address	Parameters
		One-Word	Two-Word
Direct/Immediate	0xx	Bits 9 and 10 of the M field + the A field = effective address	Second word = Immediate operand or operand storage location
Direct/Indirect	111	A field = address containing another address (either direct or indirect depending on the setting of bit 15)	Second word = effective address if bit 15 = 0; if bit 15 = 1, second word = address containing another address (either direct or indirect depending on the setting of bit 15)
NOTE: Extended-addressing instructions can be both indexed and indirect-addressed.			
Relative to P	100	A field + (the contents of the P register + 1) = effective address	Second word + (the contents of the P register + 1) = effective address*
Indexed with X	101	A field + the contents of the X register = effective address	Second word + the contents of the X register = effective address*
Indexed with B	110	A field + the contents of the B register = effective address	Second word + the contents of the B register = effective address*

* Applies only to the extended-addressing instructions.

up to 512 words in advance of the current program location.

The extended-addressing instructions are the only two-word instructions that can use relative addressing. In these instructions, the M field is specified by bits 0-2 of the first word, and the second word comprises the A field.

Indexed Addressing

Indexing is the adding of a variable, either positive or negative, to the address portion of an instruction. Indexing is useful for sequencing through tables and in base-referencing areas of memory.

Indexed addressing is specified by an M field of 101 (05) if the indexing register is

the X register. If the indexing register is the B register, the M field contains 110 (06).

The effective memory address of the operand for an indexed instruction is the sum of the contents of the A field and the contents of the indexing register.

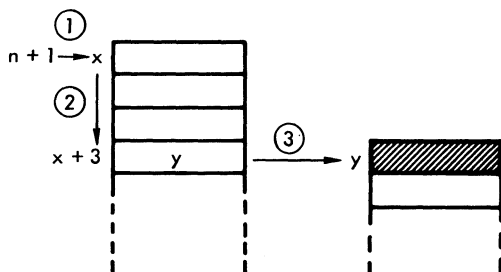
Preindexing

In the Varian 620/f-100, the extended addressing instructions can be preindexed (figure 15-1). In preindexing, the contents of the indexing register plus the contents of the second word of the instruction give an address that is, in the case of direct

PREINDEXING

$$\begin{aligned}\text{Let } (n+1) &= x \\ (B) &= 3\end{aligned}$$

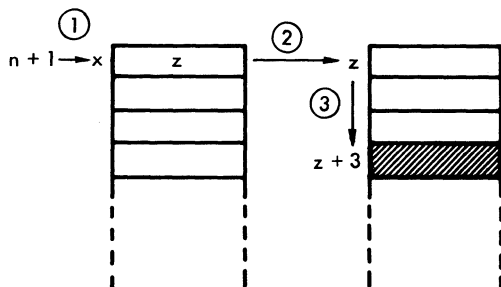
$$\begin{aligned}(n+1) + (B) &= x + 3 \\ (x + 3) &= y\end{aligned}$$



POSTINDEXING

$$\begin{aligned}\text{Let } (n+1) &= x \\ (B) &= 3\end{aligned}$$

$$\begin{aligned}(x) &= z \\ z + (B) &= z + 3\end{aligned}$$



Circled numbers indicate the sequence of events.
Shaded area is the effective memory address.

Figure 15-1. Preindexing and Postindexing

ADDRESSING MODES

addressing, the effective memory address. In the case of indirect addressing, this address is an intermediate address upon which further indirect addressing steps are based.

Thus, preindexing specifies that the indexing is performed on the contents of the second word of the instruction prior to subsequent indirect addressing steps.

Postindexing

In Varian 620/f-100 postindexing (figure 15-1), the effective memory address is found exactly as in indirect extended addressing (see below), but the final address thus found is added to the contents of the indexing register to obtain the postindexed effective memory address.

Thus, postindexing specifies that the indexing is performed on the contents of the direct address found after all indirect addressing steps have been processed. **Note that preindexing and postindexing are specified by bit 7 of the first word of the instruction; it is zero for preindexing and one for postindexing.**

Of the two-word addressing instructions, only the special-purpose extended-addressing instructions can be indexed. The M field is specified by bits 0-2 of the first word, and the second word comprises the A field.

Extended Addressing

Extended addressing permits addressing of any location up to the system maximum without using indexing or indirect addressing.

The M field (bits 0-2) of the first word of extended-addressing instructions specifies the addressing mode as defined in table 15-1. The second word of the instruction is the A field from which the effective memory address is derived (refer to the last column of table 15-1).

Immediate Addressing

Immediate-addressing instructions are actually classified as two-word nonaddressing instructions because the second word of the instruction is itself the operand. They are termed "addressing" because an operand address is implied in their structure. These instructions permit no address modification; however, they increase programming capabilities for accessing operands.

Immediate-addressing instructions are defined by mnemonics in which the last letter is I.

SECTION 16 - INSTRUCTION SET

This dictionary of Varian 620-100 series computer instructions can be divided into the following functional groups:

- Load/store instructions
- Shift/rotation instructions
- Register transfer/modification instructions
- Arithmetic instructions
- Logic instructions
- Jump instructions
- Jump-and-mark instructions
- Execution instructions
- Control instructions
- I/O instructions

Table 16-1 summarizes the length and addressing mode information applicable to these groups of instructions.

Table 16-1. Instruction Groups

Group	Length	Addressing
Load/store	One word Two words Two words	Normal Extended Immediate
Shift/rotation	One word	None
Register transfer/ modification	One word	None
Arithmetic	One word Two words Two words	Normal Extended Immediate
Logic	One word Two words Two words	Normal Extended Immediate
Jump	Two words	Extended
Jump and mark	Two words	Extended
Execution	Two words	Extended
Control	One word	None
Input/output	One word Two words	None Extended

INSTRUCTION SET

Appendix A is a list of the Varian 620-100 series instructions, arranged alphabetically by mnemonic and indexed to the page of this handbook where the instruction is explained.

Load/Store Instructions

This group comprises the instructions for loading registers from memory or for storing the contents of registers in memory. Subgroups of these instructions permit such loading or storing in normal, extended, or immediate addressing modes.

Normal Load/Store Instructions

Mnemonic

Instruction

LDA	Load A register
LDB	Load B register
LDX	Load X register
STA	Store A register
STB	Store B register
STX	Store X register

These instructions have the following one-word addressing format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Op Code					M					A					

The operation code varies for each instruction. The A field contains an address in the addressing mode specified by the M field.

LDA Load A Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
01					M					A					

Loads the contents of the effective memory address into the A register.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

LDB Load B Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
02					M					A					

Loads the contents of the effective memory address into the B register.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	B

LDX Load X Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
03					M					A					

Loads the contents of the effective memory address into the X register.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	X

STA Store A Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
05					M					A					

Stores the contents of the A register in the effective memory address.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Memory

STB Store B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

06	M	A
----	---	---

Stores the contents of the B register in the effective memory address.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Memory

STX Store X Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

07	M	A
----	---	---

Stores the contents of the X register in the effective memory address.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Memory

Extended Load/Store Instructions

This group includes:

Mnemonic	Instruction
LDAE	Load A register extended
LDBE	Load B register extended

Mnemonic Instruction

LDXE	Load X register extended
STAE	Store A register extended
STBE	Store B register extended
STXE	Store X register extended

These instructions have the following two-word addressing format:

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Op Code										M	
i	Extended Address										

These instructions have configuration 0 000 110 000 in bits 6-15. Bits 3-5 specify the manipulation. Note that the configuration of these bits is the same as the operation code for the corresponding one-word load/store instruction.

LDAE Load A Register Extended

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00601										M	
i	Extended Address										

Loads into the A register the contents of the effective memory address given by the operand address.

Timing:	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

LDBE Load B Register Extended

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00602										M	
i	Extended Address										

INSTRUCTION SET

Loads into the B register the contents of the effective memory address given by the operand address.

Timing:	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	B

LDXE Load X Register Extended

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00603														M	
i		Extended Address													

Loads into the X register the contents of the effective memory address given by the operand address.

Timing:	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	X

STAE Store A Register Extended

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00605														M	
i	Extended Address														

Stores the contents of the A register in the effective memory address given by the operand address.

Timing:	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Effective memory address

STBE Store B Register Extended

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00606														M	
i	Extended Address														

Stores the contents of the B register in the effective memory address given by the operand address.

Timing:	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Effective memory address

STXE Store X Register Extended

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00607														M	
i	Extended Address														

Stores the contents of the X register in the effective memory address given by the operand address.

Timing:	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Effective memory address

Immediate Load/Store Instructions

This group includes:

Mnemonic	Instruction
LDAI	Load A register immediate
LDBI	Load B register immediate
LDXI	Load X register immediate

Mnemonic	Instruction
STAI	Store A register immediate
STBI	Store B register immediate
STXI	Store X register immediate

These instructions have the following two-word nonaddressing format.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Op Code														M	
Operand															

These instructions have configuration 0 000 110 000 in bits 6-15. Bits 3-5 specify the manipulation. Note that the configuration of these bits is the same as that of the operation code for the corresponding one-word load/store instruction.

The M field is 000. This mode precludes relative or indirect addressing or indexing.

The second word of the instruction is always the data to be processed. It cannot be an address; in immediate addressing instructions the address is implied in the second word. Thus, in LDAI, the contents of the second word are loaded into the A register.

LDAI Load A Register Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00601														0	
Operand															

Loads into the A register the contents of the operand.

Timing:	2 cycles
Relative addressing:	No

Indirect addressing:	No
Indexing:	No
Register altered:	A

LDBI Load B Register Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00602														0	
Operand															

Loads into the B register the contents of the operand.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	B

LDXI Load X Register Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00603														0	
Operand															

Loads into the X register the contents of the operand.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	X

STAI Store A Register Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00605														0	
Operand															

INSTRUCTION SET

Stores the contents of the A register in the second word.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	Operand

STBI Store B Register Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00606														0	
Operand															

Stores the contents of the B register in the second word.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	Operand

STXI Store X Register Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00607														0	
Operand															

Stores the contents of the X register in the second word.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	Operand

Shift/Rotation Instructions

This group comprises the instructions that shift or rotate the contents of registers. In

shifts, the bits shifted out of the register are lost; but in rotation, they are loaded one at a time into the opposite end of the register.

Mnemonic	Instruction
----------	-------------

LSRA	Logical shift right A register
LSRB	Logical shift right B register
LRLA	Logical rotation left A register
LRLB	Logical rotation left B register
LLSR	Long logical shift right
LLRL	Long logical rotation left
ASRA	Arithmetic shift right A register
ASRB	Arithmetic shift right B register
ASLA	Arithmetic shift left A register
ASLB	Arithmetic shift left B register
LASR	Long arithmetic shift right
LASL	Long arithmetic shift left

These instructions have the following one-word nonaddressing format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Op Code															

These instructions have configuration 0 000 100 in bits 9-15. Bits 0-4 specify the number of bits to be shifted/rotated. Bits 5-8 specify the parameters of the shift/rotation operation, as follows:

Bit 5 = 0	Arithmetic shift/rotation
Bit 5 = 1	Logical shift/rotation

Bit 6 = 0 Left shift/rotation
 Bit 6 = 1 Right shift/rotation

Bit 7 = 0 B register shift/
 rotation

Bit 7 = 1 A register shift/
 rotation

Bit 8 = 0 Shift/rotate one reg-
 ister only

Bit 8 = 1 Long shift/rotation

Rotations occur when bit 5 = 1 and bit 6 = 0; otherwise, there is a shift. Note that for long shifts/rotations, bit 7 = 0.

LSRA Logical Shift Right A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004340 + x

Shifts the contents of the A register x places ($x = 0$ to 378) to the right and loads the vacated high-order bit(s) with zeros. Information shifted out of the low-order bit(s) is lost.

Timing: f-100: $1 + 0.147(x - 1)$ cycles
 L-100: $1 + 0.250(x - 1)$ cycles

Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: A

LSRB Logical Shift Right B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004140 + x

Shifts the contents of the B register x places ($x = 0$ to 378) to the right and loads the vacated high-order bit(s) with zeros. Information shifted out of the low-order bit(s) is lost.

Timing: f-100: $1 + 0.147(x - 1)$ cycles
 L-100: $1 + 0.250(x - 1)$ cycles

Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: B

LRLA Logical Rotate Left A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004240 + x

Rotates the contents of the A register x places ($x = 0$ to 378) to the left. Bit 0 receives each high-order bit as it is rotated out during the execution.

Timing: f-100: $1 + 0.147(x - 1)$ cycles
 L-100: $1 + 0.250(x - 1)$ cycles

Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: A

LRLB Logical Rotation Left B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004040 + x

Rotates the contents of the B register x places ($x = 0$ to 378) to the left. Bit 0 receives each high-order bit as it is rotated out during the execution.

Timing: f-100: $1 + 0.147(x - 1)$ cycles
 L-100: $1 + 0.250(x - 1)$ cycles

Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: B

INSTRUCTION SET

LLSR Long Logical Shift Left

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004540 + x

Shifts the contents of the A and B registers x places ($x = 0$ to 037) to the right. Loads the vacated high-order bit(s) of the A register with zeros. Information shifted out of the low-order bit(s) of the B register is lost.

Timing: f-100: $1 + 0.147(x - 1)$ cycles
L-100: $1 + 0.250(x - 1)$ cycles

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A and B

LLRL Long Logical Rotation Right

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004440 + x

Rotates the contents of the A and B registers x places ($x = 0$ to 037). Bit 0 of the B register receives each high-order bit of the A register as it is rotated out during the execution.

Timing: f-100: $1 + 0.147(x - 1)$ cycles
L-100: $1 + 0.250(x - 1)$ cycles

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A and B

ASRA Arithmetic Shift Right A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004300 + x

Shifts the contents of the A register, including the sign bit, x places ($x = 0$ to 037) to the right. Loads the vacated high-order bit(s) with the value of the sign bit. Information shifted out of the low-order bit(s) is lost.

Timing: f-100: $1 + 0.147(x - 1)$ cycles
L-100: $1 + 0.250(x - 1)$ cycles

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A

ASRB Arithmetic Shift Right B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004100 + x

Shifts the contents of the B register, including the sign bit, x places ($x = 0$ to 037) to the right. Loads the vacated high-order bit(s) with the value of the sign bit. Information shifted out of the low-order bit(s) is lost.

Timing: f-100: $1 + 0.147(x - 1)$ cycles
L-100: $1 + 0.250(x - 1)$ cycles

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: B

**ASLA Arithmetic Shift Left
 A Register**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

$$004200 + x$$

Shifts the contents of the A register x places ($x = 0$ to 037) to the left. Loads the vacated low-order bit(s). Information shifted out of the high-order bit(s) is lost, but the sign bit is unaffected.

Timing: f-100: $1 + 0.147 (x - 1)$ cycles
L-100: $1 + 0.250 (x - 1)$ cycles

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

ASLB **Arithmetic Shift Left**
 B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

$$004000 + x$$

Shifts the contents of the B register x places ($x = 0$ to 037) to the left loads the vacated low-order bit(s) with zero(s). Information shifted out of the high-order bit(s) is lost, but the sign bit is unaffected.

Timing: f-100: $1 + 0.147 (x - 1)$ cycles
L-100: $1 + 0.250 (x - 1)$ cycles

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	B

LASR **Long Arithmetic Shift Right**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004500 + x

The contents of the A register are shifted n positions to the right ($n = 0$ to 037). Bits shifted out of the low-order positions of A are lost. The sign bit (bit 15) of the A register is extended n places to the right.

Timing: f-100: $1 + 0.147 (x - 1)$ cycles
L-100: $1 + 0.250 (x - 1)$ cycles

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A and B

LASL **Long Arithmetic Shift Left**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004400 + x

The contents of the A and B registers are shifted n places to the left ($n = 0$ to 037). Bit position 14 of the B register is shifted into bit position 0 of the A register; the sign bit (bit 15) of the B register remains unchanged. The sign bit (bit 15) of the A register also remains unchanged. The bit shifted out of bit position 14 of the A register is lost, and zeros are shifted into the low-order positions of the B register.

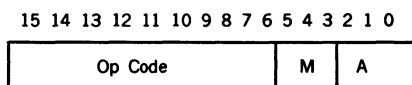
Timing: f-100: $1 + 0.147 (x - 1)$ cycles
L-100: $1 + 0.250 (x - 1)$ cycles

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A and B

Register Transfer/Modification Instructions

This group comprises the unmodified register transfers; instructions for incrementing, decrementing, and complementing registers and for adjusting the contents of registers with the overflow indicator; and microcoded combinations of these operations.

These instructions have the following one-word pseudoeaddressing format:



This group of instructions does not address memory. However, this format is included in the one-word addressing grouping because registers are, in effect, addressed and their contents modified by the operations, much as in operations involving memory.

These instructions have configuration 000 101 in bits 9-15.

Bit 8 specifies conditional execution of the instruction depending upon the overflow indicator:

0cc	Execute instruction unconditionally
1cc	Execute only if the overflow indicator is set

Bits 6 and 7 specify that data be transferred from one register to another as follows:

w00	Unmodified
w01	Incremented
w10	Complemented
w11	Decrement

Bits 3-5 specifies the location from which the data to be transferred or modified are obtained (source register):

000	Clears destination register
001	A register
010	B register
100	X register

Bits 0-2 specify the location in which the modified and/or transferred data are placed (destination register):

000	No operation
001	A register
010	B register
100	X register

Additional transfers and modifications can be microcoded. Thus, 110 in bits 0-2 places data in both the X and B registers. If bits 3-5 specify more than one source register, the result is the inclusive-OR of the group of registers.

The instructions of this group are summarized immediately preceding the appropriate subgrouping.

Unmodified Register Transfer

Instructions

Mnemonic	Instruction
TAB	Transfer A register to B register
TAX	Transfer A register to X register
TBA	Transfer B register to A register
TBX	Transfer B register to X register
TXA	Transfer X register to A register
TXB	Transfer X register to B register

TZA Transfer zeros to A register (clear A)
TZB Transfer zeros to B register (clear B)
TZX Transfer zeros to X register (clear X)
TSA Transfer switches to A register

TAB Transfer A Register to B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0050	1	2
------	---	---

Transfers the contents of the A register to the B register.

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: B

TAX Transfer A Register to X Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0050	1	4
------	---	---

Transfers the contents of the A register to the X register.

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: X

TBA Transfer B Register to A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0050	2	1
------	---	---

Transfers the contents of the B register to the A register.

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: A

TBX Transfer B Register to X Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0050	2	4
------	---	---

Transfers the contents of the B register to the X register.

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: X

TXA Transfer X Register to A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0050	4	1
------	---	---

Transfers the contents of the X register to the A register.

Timing: 1 cycle
 Relative addressing: No

INSTRUCTION SET

Indirect addressing: No
Indexing: No
Register altered: A

TXB Transfer X Register to B Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0050												4	2		

Transfers the contents of the X register to the B register.

Timing: 1 cycle
Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: B

TZA Transfer Zeros to A Register (Clear A)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0050												0	1		

Clears the A register.

Timing: 1 cycle
Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A

TZB Transfer Zeros to B Register (Clear B)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0050												0	2		

Clears the B register.

Timing: 1 cycle
Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: B

TZX Transfer Zeros to X Register (Clear X)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0050												0	4		

Clears the X register.

Timing: 1 cycle
Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: X

TSA Transfer Switches to A Register*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
007402															

Transfers the contents of the register entry switches to the A register.

Timing: 1 cycle
Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A

* Applicable to the Varian 620/f-100 only.

Register Modification Instructions

Mnemonic Instruction

IAR Increment A register
IBR Increment B register

INSTRUCTION SET

IXR	Increment X register
DAR	Decrement A register
DBR	Decrement B register
DXR	Decrement X register
CPA	Complement A register
CPB	Complement B register
CPX	Complement X register
AOFA	Increment A register if overflow indicator set
AOFB	Increment B register if overflow indicator set
AOFX	Increment X register if overflow indicator set
SOFA	Decrement A register if overflow indicator set
SOFB	Decrement B register if overflow indicator set
SOFX	Decrement X register if overflow indicator set

Note in the following examples that one instruction in each grouping shows the overflow conditional execution that is also applicable to other instructions in the group.

IXR Increment X Register Unconditionally

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0051												1	1		

IBR Increment B Register if Overflow Indicator Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0055												2	2		

IXR Increment X Register Unconditionally

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0051												4	4		

Increments (by one) the contents of the specified register. Sets the overflow indicator (OF) if the register contents reach the maximum positive number (077777); changes the contents to the maximum negative number (0100000).

Timing:	1 cycle
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	OF and that specified by the second letter of the mnemonic

DAR Decrement A Register Unconditionally

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0053												1	1		

DBR Decrement B Register Unconditionally

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0053												2	2		

INSTRUCTION SET

DXR Decrement X Register if Overflow Indicator Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0057												4	4		

Decrements (by one) the contents of the specified register. Sets OF if the register contents are 0100000 when executed and changes the contents to 077777.

Timing:	1 cycle
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	OF and that specified by the second letter of the mnemonic

CPA Complement A Register if Overflow Indicator Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0056										1	1				

CPB Complement B Register Unconditionally

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0052										2	2				

CPX Complement X Register Unconditionally

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0052											4	4			

One's-complements the contents of the specified register.

Timing:	1 cycle
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	That specified by the third letter of the mnemonic

AOFA Increment A Register if Overflow Indicator Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0055											1	1			

AOFB Increment B Register if Overflow Indicator Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0055											2	2			

AOFX Increment X Register if Overflow Indicator Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0055											4	4			

Adds one to the contents of the specified register only if OF is set. Does not change the setting of OF.

Subtracts one from the contents of the specified register only if OF is set. Does not change the setting of OF.

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: That specified by the fourth letter of the mnemonic

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: That specified by the fourth letter of the mnemonic

SOFA Decrement A Register if Overflow Indicator Set

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0057	1	1
------	---	---

SOFB Decrement B Register if Overflow Indicator Set

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0057	2	2
------	---	---

SOFX Decrement X Register if Overflow Indicator Set

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0057	4	4
------	---	---

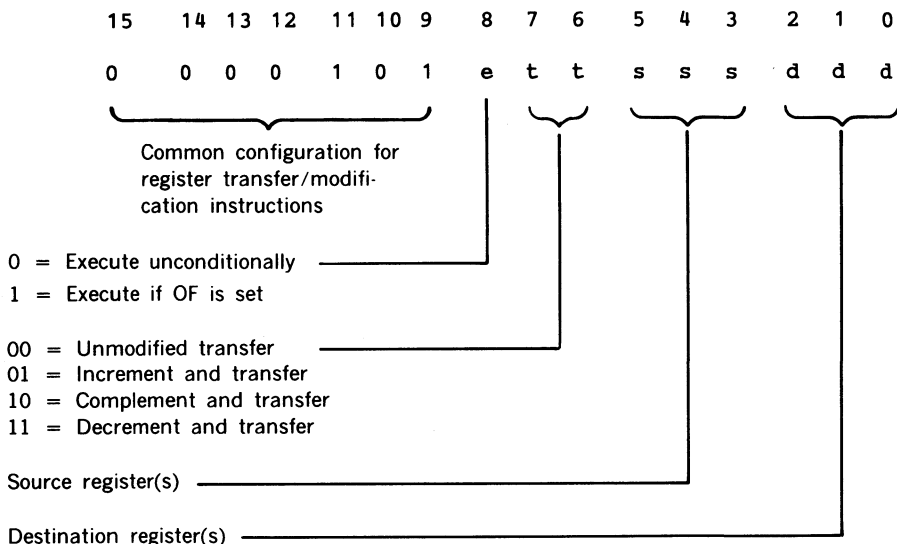
Combined Register Transfer/Modification Instructions

These instructions are used to specify combinations of the register transfer and modification instructions to perform simultaneous multiple operations. The combined conditions are established in the variable field of the DAS assembler statement (section 17) when the program is written.

Mnemonic	Instruction
MERG	Merge source to destination registers
INCR	Increment source to destination registers
DECR	Decrement source to destination registers
COMPL	Complement source to destination registers
ZERO	Zero (clear) registers

INSTRUCTION SET

The instruction bits specify the parameters indicated below.



Bits 3-5 specify the source register(s) as follows:

- 000 Clears destination register(s)
- 001 A register
- 010 B register
- 100 X register

Thus, a configuration of 110 specifies that the contents of the B and X registers are to be inclusively ORed before the transfer/modification operation is performed.

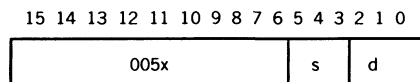
Bits 0-2 specify the destination register(s) as follows:

- 000 No operaton
- 001 A register
- 010 B register
- 100 X register

Thus, a configuration of 011 places data in both the A and B registers when the instructions is executed.

MERG

Merge Source to Destination Registers

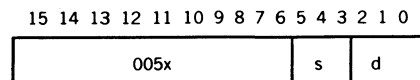


Transfers the inclusive-OR of the contents of the source register(s) to the destination register(s).

Timing:	1 cycle
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Registers altered:	Those specified

INCR

Increment Source to Destination Registers



Adds one to the inclusive-OR of the contents of the source register(s), and places the result in the destination register(s).

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Registers altered: Those specified

DECR Decrement Source to Destination Registers

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

005x	s	d
------	---	---

Subtracts one from the inclusive-OR of the contents of the source register(s), and places the result in the destination register(s).

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Registers altered: Those specified

COMPL Complement Source to Destination Registers

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

005x	s	d
------	---	---

One's-complements the inclusive-OR of the contents of the source register(s), and places the result in the destination register(s).

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Registers altered: Those specified

ZERO

Zero (Clear) Registers

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

005x	s	d
------	---	---

Clears the destination register(s).

Timing: 1 cycle
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Registers altered: Those specified

Arithmetic Instructions

This group comprises the instructions for incrementation of the contents of a memory address for performing the arithmetic functions of addition, subtraction, multiplication, and division; and the extended- and immediate-addressing counterparts of these instructions. Multiplication and division are optional.

Mnemonic	Instruction
INR	Increment memory and replace
INRE	Increment memory and replace extended
INRI	Increment memory and replace immediate
ADD	Add memory to A register
ADDE	Add memory to A register extended
ADDI	Add memory to A register immediate
SUB	Subtract memory from A register
SUBE	Subtract memory from A register extended
SUBI	Subtract memory from A register immediate

continued

MUL	Multiply
MULE	Multiply extended
MULI	Multiply immediate
DIV	Divide
DIVE	Divide extended

In the one-word instructions, bits 12-15 specify the arithmetic function:

0	100	Increment
1	010	Add
1	100	Subtract
1	110	Multiply
1	111	Divide

In the two-word instructions, the same configurations appear in bits 3-6 of the first word.

INR Increment Memory and Replace

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

04	M	A
----	---	---

Increments (by one) the contents of the effective memory address. Sets the overflow indicator (OF) if the maximum positive number (077777) is exceeded. The value in the memory address is then 010000.

Timing:	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Registers altered:	Memory and OF

INRE Increment Memory and Replace Extended

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00604	M
i	Extended Address

Increments (by one) the contents of the effective memory address designated by the operand in the second word and sets OF as for INR.

Timing:	4 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Registers altered:	Memory and OF

INRI Increment and Replace Immediate

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00604	0
Operand	

Increments (by one) the operand in the second word and sets OF as for INR.

Timing:	3 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Registers altered:	Memory and OF

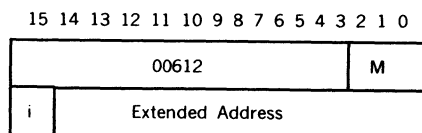
ADD Add Memory to A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

12	M	A
----	---	---

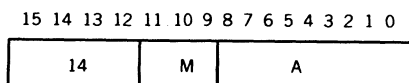
Adds the contents of the effective memory address to the contents of the A register, places the sum in the A register, and sets OF and sign bit 15 to one if 077777 is exceeded.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A and OF

**ADDE Add Memory to A Register
Extended**


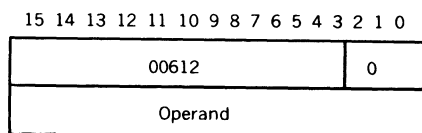
Similar to ADD using as adds the contents of the A register and the contents of the effective memory address designated by the operand address in the second word.

Timing: 3 cycles
 Relative addressing: Yes
 Indirect addressing: Yes
 Indexing: Yes
 Register altered: A and OF

**SUB Subtract Memory From
A Register**


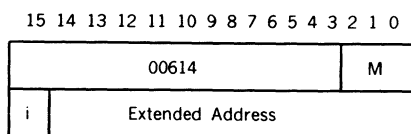
Subtracts the contents of the effective memory from the contents of the A register, places the remainder in the A register, and sets OF and resets sign bit 15 if the result exceeds the maximum negative number (0100000).

Timing: 2 cycles
 Relative addressing: Yes
 Indirect addressing: Yes
 Indexing: Yes
 Register altered: A and OF

ADDI Add to A Register Immediate


Similar to ADD using as adds the contents of the A register and the operand in the second word.

Timing: 2 cycles
 Relative addressing: No
 Indirect addressing: No
 Indexing: No
 Register altered: A and OF

**SUBE Subtract Memory From
A Register Extended**


Similar to SUB using as subtrahend the effective memory address designated by the operand address in the second word.

Timing: 3 cycles
 Relative addressing: Yes
 Indirect addressing: Yes
 Indexing: Yes
 Register altered: A and OF

INSTRUCTION SET

SUBI Subtract From A Register Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00614														0	
Operand															

Similar to SUB using as subtrahend the operand in the second word.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A and OF

MUL Multiply

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
16				M				A							

Multiplies the contents of the effective memory address by the contents of the B register, adds the contents of the A register to the product, and places the result in the A and B registers with the most significant portion in the A register. The sign bit of the A register gives the sign of the result. The sign bit of the B register is reset to zero. OF sets if the contents of the B register and the effective memory address are the greatest possible negative number. In this case, the A and B registers contain a negative sign and zeros.

The algorithm is: $R \cdot B + A$ A,B.

Timing:	f-100: 8.5 cycles L-100: 10 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A, B, and OF

MULE Multiply Extended

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00616														M	
i	Extended Address														

Similar to MUL except that the multiplicand is the contents of the effective memory address designated by the operand and address in the second word.

Timing:	f-100: 9.5 cycles L-100: 11 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A, B, and OF

MULI Multiply Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00616														0	
Operand															

Similar to MUL using as multiplicand the operand in the second word.

Timing:	f-100: 8.5 cycles L-100: 10 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A, B, and OF

DIV Divide

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
17				M				A							

Divides the combined contents of the A and B registers by the contents of the effective memory address, and places the signed quotient in the B register and the

remainder (with the sign of the dividend) in the A register. Sets OF if necessary indicating an invalid and unpredictable quotient; overflow cannot occur if the divisor is greater than the portion of the dividend that is in the A register.

Timing: f-100: 8.5 cycles
L-100: 10 cycles

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A, B, and OF

DIVE Divide Extended

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00617													M
i	Extended Address												

Similar to DIV using as divisor the contents of the effective memory address designated by the operand address in the second word.

Timing: f-100: 9.5 cycles
L-100: 11 cycles

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A, B, and OF

DIVI Divide Immediate

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00517													0
Operand													

Similar to DIV using as divisor the operand in the second word.

Timing: f-100: 8.5 cycles
L-100: 10 cycles

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A, B, and OF

Logic Instructions

This group comprises the inclusive-OR, exclusive-OR, and AND instructions and their extended- and immediate-addressing counterparts.

Mnemonic Instruction

ORA	Inclusive-OR memory and A register
ORAE	Inclusive-OR extended
ORAI	Inclusive-OR immediate
ERA	Exclusive-OR memory and A register
ERAE	Exclusive-OR Extended
ERAI	Exclusive-OR immediate
ANA	AND memory and A register
ANAE	AND extended
ANAI	AND immediate
SRE	Skip if register equal

In the one-word instructions, bits 12-15 specify the logic function:

1	001	Inclusive-OR
1	011	Exclusive-OR
1	101	AND

In the two-word instructions, the same configurations appear in bits 3-6 of the first word.

ORA Inclusive-OR Memory and A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

11	M	A
----	---	---

INSTRUCTION SET

Performs an inclusive-OR between each bit of the A register and the corresponding bit of the effective memory address, and places the result in the A register according to the truth table for inclusive-OR shown below. In the truth table, n = bit position.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

Inclusive-OR Truth Table

Condition		Result
A(n)	Effective Memory Address (n)	A(n)
0	0	0
0	1	1
1	0	1
1	1	1

ORAE Inclusive-OR Extended

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	
00611	M
Extended Address	

Performs an inclusive-OR between each bit of the A register and the corresponding bit of the effective memory address given in the second word, and places the result in the A register according to the truth table above.

Timing:	2 cycles
Relative addressing:	Yes

Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

ORAI Inclusive-OR Immediate

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	
00611	0
Operand	

Performs an inclusive-OR between each bit of the A register and the corresponding bit of the operand, in the second word, and the result in the A register according to the truth table above.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

ERA Exclusive-OR Memory and A Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
13				M		A									

Performs an exclusive-OR between each bit of the A register and the corresponding bit of the effective memory address, and places the result in the A register according to the truth table for exclusive-OR shown below. In the truth table, n = bit position.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

Exclusive-OR Truth Table

Condition		Result
A(n)	Effective Memory Address (n)	A(n)
0	0	0
0	1	1
1	0	1
1	1	0

places the result in the A register according to the truth table above.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

ANA**AND Memory and A register****ERAE****Exclusive-OR Extended**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00613	M
Extended Address	

Performs an exclusive-OR between each bit of the A register and the corresponding bit of the effective memory address given in the second word, and places the result in the A register according to the truth table above.

Timing:	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

ERAI**Exclusive-OR Immediate**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00613	0
Operand	

Performs an exclusive-OR between each bit of the A register and the corresponding bit of the operand in the second word, and

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

15	M	A
----	---	---

Performs an AND between each bit of the A register and the corresponding bit of the effective memory address, and places the result in the A register according to the truth table for AND shown below. In the truth table, n = bit position.

Timing:	2 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

AND Truth Table

Condition		Result
A(n)	Effective Memory Address (n)	A(n)
0	0	0
0	1	0
1	0	0
1	1	1

INSTRUCTION SET

ANAE

AND Extended

SRE

Skip if Register Equal *

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00615	M
Extended Address	

Performs an AND between each bit of the A register and the corresponding bit of the effective memory address given in the second word, and places the result in the A register according to the truth table above.

Timing:	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

ANAI

AND Immediate

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00615	0
Operand	

Performs an AND between each bit of the A register and the corresponding bit of the operand in the second word, and places the result in the A register according to the truth table above.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0066	x	M
i	Operand Address	

Makes a logical comparison between the register specified by bits 3-5 and the word at the address specified by the second word of the instruction. Bits 3-5 = 001 specifies the A register, bits 3-5 = 010 specifies the B register, and bits 3-5 = 100 specifies the X register.

Bits 0-2 specify the addressing mode: 000 or 100 = relative to P, 001, or 101 = indexed with X, 010 or 110 = indexed with B, and 011 or 111 = direct/indirect (postindexing).

If the compared quantities are equal, the program skips the next two locations and executes the instruction in the third location. If the compared quantities are unequal, the program executes the instruction immediately following SRE.

Timing:	
Condition met	3.5 cycles
Not met	3 cycles
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Postindexing
Register altered:	None

* Applicable to the Varian 620/f-100 only.

Jump Instructions

This group comprises the instructions that direct the program to a nonsequential address for execution of the instruction

located there, but they neither mark the location (as do the jump-and-mark instructions) nor do they bring the program back to the main program sequence (as do the execution instructions).

Mnemonic**Instruction**

JMP	Jump unconditionally
IJMP	Indexed jump
JOF	Jump if overflow indicator set
JOFN	Jump if overflow indicator not set
JAP	Jump if A register positive
JAN	Jump if A register negative
JAZ	Jump if A register zero
JBZ	Jump if B register zero
JXZ	Jump if X register zero
JANZ	Jump if A register not zero
JBNZ	Jump if B register not zero
JXNZ	Jump if X register not zero
JSS1	Jump if SENSE switch 1 set
JSS2	Jump if SENSE switch 2 set
JSS3	Jump if SENSE switch 3 set
JS1N	Jump if SENSE switch 1 not set
JS2N	Jump if SENSE switch 2 not set
JS3N	Jump if SENSE switch 3 not set
JIF	Jump if condition(s) met
JSR	Jump and return in indexing register
BT	Bit test

These instructions have the following two-word addressing format:

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Op Code										M									
i	Jump Address																		

This format comprises a seven-bit operation code (bits 9-15), a nine-bit M field (bits 0-8) in the first word, and an effective jump address in the second.

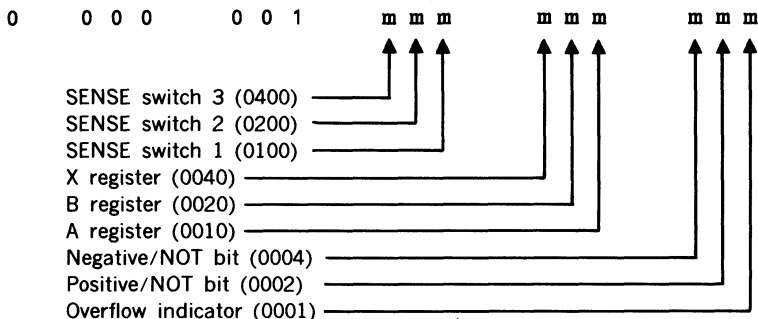
These instructions have configuration 0 000 001 in bits 9-15. Bits 0-8 specify the jump as follows.

All M field bits are zeros for an unconditional jump instruction. For other instructions in this group, each switch, register, or indicator to be examined is represented by a specific M field bit. Bits 1 and 2 specify the condition of the examined switch, register, or indicator to be met for a jump. If bits 1 and 2 are zero, the jump condition is met if the switch or indicator is set, or if the register contains all zeros. If bits 1 and 2 are one, the jump condition is met if the switch or indicator is not set, or if the register contains any number but positive zero.

If only bit 1 of the M field is set, the jump condition is met when the A register contains a positive value (bit 15 off). If only bit 2 is set, the jump condition is met when the A register contains a negative value (bit 15 set). Note that, in these two cases, the A register bit (bit 3) is zero. Bit 15 in the other registers is not always an arithmetic sign. Therefore, setting only bit 1 or bit 2 does not affect the B or X register since there are no analogous instructions for these registers.

M field bits specify, when set, the parameters indicated below:

INSTRUCTION SET

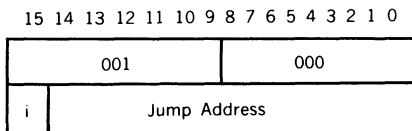


Refer to the discussion of assembler instruction types in section 17 for details of microcoding multiple instruction conditions.

In these instructions, the effective jump address in the second word is the address of the next instruction to be executed if the jump condition is met. The program then continues to execute instructions following the jump address.

If the jump condition is not met, the program executes the instruction immediately following the second word of the jump instruction.

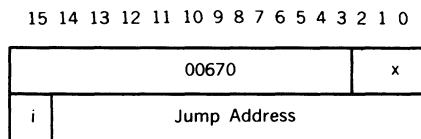
JMP Jump Unconditionally



Jumps unconditionally to the instruction at the effective jump address and executes it next.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

IJMP Indexed Jump*



Jumps unconditionally to the instruction at the effective postindexed jump address and executes it next.

The effective postindexed jump address is formed by indexing the effective jump address given in the second word with the contents of the register specified by bits 0-2.

If bits 0-2 = 101, the indexing register is the X register; if bits 0-2 = 110, the indexing register is the B register.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	Postindexing
Register altered:	P

* Applicable to the Varian 620/f-100 only.

JOF **Jump if Overflow Indicator Set**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001										001									
i		Jump Address																	

If the overflow indicator is set, jumps to the instruction at the effective jump address and executes it next. Resets the overflow indicator.

If the overflow indicator is not set, executes the next instruction in sequence.

Timing:

f-100: 1 or 2 cycles

L-100: 2 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P and OF

JOFN **Jump if Overflow Indicator Not Set***

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001										007									
i	Jump Address																		

If the overflow indicator is not set, jumps to the instruction at the effective jump address and executes it next. If the overflow indicator is set, executes the next instruction in sequence.

Timing: 1 or 2 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Registers altered: OF and P

* Applicable to the Varian 620/f-100 only.

JAP **Jump if A Register Positive**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001										002									
i		Jump address																	

If the A register contains a positive value (including zero), jumps to the instruction at the effective jump address and executes it next. If the A register contains a negative value, executes the next instruction in sequence.

Timing:

f-100: 1 or 2 cycles

L-100: 2 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P

JAN **Jump if A Register Negative**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001										004									
i		Jump Address																	

If the A register contains a negative value, jumps to the instruction at the effective jump address and executes it next. If the A register contains a positive value (including zero), executes the next instruction in sequence.

Timing:

f-100: 1 or 2 cycles

L-100: 2 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P

INSTRUCTION SET

JAZ Jump if A Register Zero

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001										010									
i		Jump Address																	

If the A register contains zero, jumps to the instruction at the effective jump address and executes it next. If the A register does not contain zero, executes the next instruction in sequence.

Timing:

f-100: 1 or 2 cycles

L-100: 2 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P

JBZ Jump if B Register Zero

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001										020									
i	Jump Address																		

If the B register contains zero, jumps to the instruction at the effective jump address and executes it next. If the B register does not contain zero, executes the next instruction in sequence.

Timing:

f-100: 1 or 2 cycles

L-100: 2 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P

JXZ Jump if X Register Zero

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001										040									
i		Jump Address																	

If the X register contains zero, jumps to the instruction at the effective jump address and executes it next. If the X register does not contain zero, executes the next instruction in sequence.

Timing:

f-100: 1 or 2 cycles

L-100: 2 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P

JANZ Jump if A Register Not Zero*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001										016									
i	Jump Address																		

If the A register is not zero, executes next the instruction at the jump address. If the A register is zero, executes the next instruction in sequence.

Timing:

1 or 2 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P

* Applicable to the Varian 620/f-100 only.

JBNZ Jump if B Register Not Zero*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001										026									
i										Jump Address									

If the B register is not zero, executes next the instruction at the jump address. If the B register is zero, executes the next instruction in sequence.

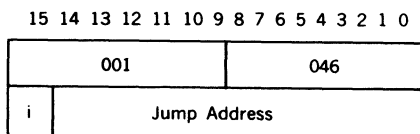
Timing: 1 or 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: P

not set, executes the next instruction in sequence.

Timing:
 f-100: 1 or 2 cycles
 L-100: 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: P

* Applicable to the Varian 620/f-100 only.

JXNZ Jump if X Register Not Zero*

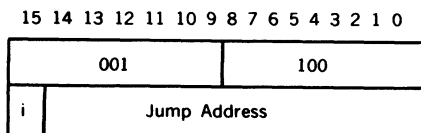


If the X register is not zero, executes next the instruction at the jump address. If the X register is zero, executes the next instruction in sequence.

Timing: 1 or 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: P

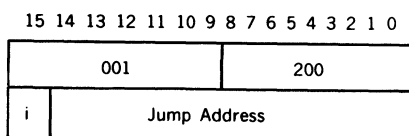
* Applicable to the Varian 620/f-100 only.

JSS1 Jump if SENSE Switch 1 Set



If SENSE switch 1 is set, jumps to the instruction at the effective jump address and executes it next. If SENSE switch 1 is

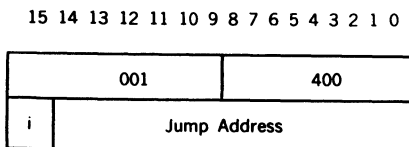
JSS2 Jump if Sense Switch 2 Set



If SENSE switch 2 is set, jumps to the instruction at the effective jump address and executes it next. If SENSE switch 2 is not set, executes the next instruction in sequence.

Timing:
 f-100: 1 or 2 cycles
 L-100: 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: P

JSS3 Jump if SENSE Switch 3 Set



If SENSE switch 3 is set, jumps to the instruction at the effective jump address and executes it next. If SENSE switch 2 is not set, executes the next instruction in sequence.

INSTRUCTION SET

Timing:	
f-100:	1 or 2 cycles
L-100:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

JS1N Jump if SENSE Switch 1
 Not Set*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001		106
i	Jump Address	

If SENSE switch 1 is not set, executes next the instruction at the jump address. If SENSE switch 1 is set, executes the next instruction in sequence.

Timing:	1 or 2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

* Applicable to the Varian 620/f-100 only.

JS2N **Jump if SENSE Switch 2
Not Set***

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001		206	
i	Jump Address		

If SENSE switch 2 is not set, executes next the instruction at the jump address. If

SENSE switch 2 is set, executes the next instruction in sequence.

Timing:	1 or 2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

* Applicable to the Varian 620/f-100 only.

JS3N **Jump if SENSE Switch 3**
 Not Set*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001		406	
i	Jump Address		

If SENSE switch 3 is not set, executes next the instruction at the jump address. If SENSE switch 3 is set, executes the next instruction in sequence.

Timing:	1 or 2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

* Applicable to the Varian 620/f-100 only.

JIF **Jump if Condition(s) Met**

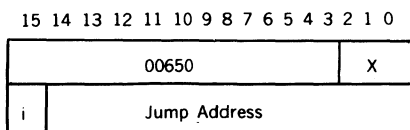
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

001		M
i	Jump Address	

If all the conditions specified by bits 0-8 are met, jumps to the instruction at the effective jump address and executes it next. The condition specified by setting combinations of M field bits is the AND of each bit specification as given in the preceding nine instructions (excluding JMP). JIF is used to microcode such combined conditions.

JSR

**Jump and Set Return
in Indexing Register***



M Field			Code	Jump Condition
000	000	001	0001	Overflow indicator set
000	000	010	0002	A register contents ≥ 0
000	000	110	0006	Jump if specified condition is NOT met.
000	000	100	0004	A register contents < 0
000	001	000	0010	A register contents = 0
000	010	000	0020	B register contents = 0
000	100	000	0040	X register contents = 0
001	000	000	0100	SENSE switch 1 set
010	000	000	0200	SENSE switch 2 set

Compound conditions are specified in the first expression of the DAS assembler variable field (section 17).

Note that some combinations are impossible, e.g., jump if overflow indicator is not set and the A register positive (because of the conflicting use of bit 1).

If not all the jump conditions are met, JIF executes the next instruction in sequence.

Stores the contents of the P register in the indexing register specified by bits 0-2, then jumps unconditionally to the instruction at the effective jump address and executes it next.

If bits 0-2 = 101, the return register is the X register; if bits 0-2 = 110, the return register is the B register.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	P and B or X

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	Those specified

* Applicable to the Varian 620/f-100 only.

INSTRUCTION SET

BT Bit Test *

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

0064										Z										
i	Jump Address																			

Tests the condition of a selected bit in the A or B register, and jumps if the condition is met. Bits 0-3 select the bit to be tested. Bits 4-5 define the condition to be met as follows:

If bits 4-5 =	BT jumps when:
00	The selected bit of the A register is 1
01	The selected bit of the B register is 1
10	The selected bit of the A register is 0
11	The selected bit of the B register is 0

If the specified condition is not met, the program executes the next instruction in sequence.

Timing:

Condition met	2 cycles
Not met	1 cycle
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

* Applicable to the Varian 620/f-100 only.

Jump-and-Mark Instructions

This group comprises the instructions that direct the program to a nonsequential jump address, store the contents of the P

register there, and next execute the instruction following the jump address.

Mnemonic

Instruction

JMPM	Jump and mark unconditionally
JOFM	Jump and mark if overflow indicator set
JOFNM	Jump and mark if overflow indicator not set*
JAPM	Jump and mark if A register positive
JANM	Jump and mark if A register negative
JAZM	Jump and mark if A register zero
JBZM	Jump and mark if B register zero
JXZM	Jump and mark if X register zero
JANZM	Jump and mark if A register not zero *
JBNZM	Jump and mark if B register not zero *
JXNZM	Jump and mark if X register not zero *
JS1M	Jump and mark if SENSE switch 1 set
JS2M	Jump and mark if SENSE switch 2 set
JS3M	Jump and mark if SENSE switch 3 set
JS1NM	Jump and mark if SENSE switch 1 not set *
JS2NM	Jump and mark if SENSE switch 2 not set *
JS3NM	Jump and mark if SENSE switch 3 not set *
JIFM	Jump and mark if condition(s) met

* Applicable to the Varian 620/f-100 only.

These instructions have the same two-word addressing format as the jump instructions.

The operation code in bits 9-15 of these instructions has the configuration 0 000 010.

The M field (bits 0-8) has the same significance as the equivalent jump instruction.

The effective jump address in the second word is the address where the contents of the P register are stored if the jump-and-mark condition is met. The instruction next to be executed is located in the jump address plus one. The program then continues to execute instructions following the one in the jump address plus one.

If the jump-and-mark condition is not met, the program executes the instruction following the jump-and-mark instruction.

JMPM Jump and Mark Unconditionally

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										000									
i		Jump Address																	

Jumps unconditionally to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address.

Timing:

f-100:	3 cycles
L-100:	2 or 3 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

JOFM Jump and Mark if Overflow Indicator Set

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										001									
i		Jump Address																	

If the overflow indicator is set, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. Resets the overflow indicator. If the overflow indicator is not set, executes the next instruction in sequence.

Timing:

f-100:	1 or 3 cycles
L-100:	2 or 3 cycles

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P, OF, and effective jump address

JOFNM Jump and Mark if Overflow * Indicator Not Set

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										007									
i		Jump Address																	

If the overflow indicator is not set, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If the overflow indicator is set, executes the next instruction in sequence. Does not reset OF.

Timing:

Condition met:	3 cycles
Condition not met:	1 cycle
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

* Applicable to the Varian 620/f-100 only.

INSTRUCTION SET

JAPM Jump and Mark if A Register Positive

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										002									
i	Jump Address																		

If the A register contains a positive value (including zero), jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the A register contains a negative value, executes the next instruction in sequence.

Timing:
f-100: 1 or 3 cycles
L-100: 2 or 3 cycles
Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P and effective jump address

JANM Jump and Mark if A Register Negative

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002		004	
i	Jump Address		

If the A register contains a negative value, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the A register contains a positive value (including zero), executes the next instruction in sequence.

Timing:
f-100: 1 or 3 cycles
L-100: 2 or 3 cycles
Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P and effective jump address

JAZM Jump and Mark if A Register Zero

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002		010	
i	Jump address		

If the A register contains zero, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the A register does not contain zero, executes the next instruction in sequence.

Timing:
f-100: 1 or 3 cycles
L-100: 2 or 3 cycles
Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P and effective jump address

JBZM Jump and Mark if B Register Zero

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										020									
i	Jump Address																		

If the B register contains zero, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the B register does not contain zero, executes the next instruction in sequence.

Timing:

f-100: 1 or 3 cycles

L-100: 2 or 3 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P and effective jump address

JXZM Jump and Mark if X Register Zero

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										040									
i	Jump Address																		

If the X register contains zero, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the X register does not contain zero, executes the next instruction in sequence.

Timing:

f-100: 1 or 3 cycles

L-100: 2 or 3 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P and effective jump address

JANZM Jump and Mark if A Register Not Zero*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										016									
i	Jump Address																		

If the A register is not zero, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If the A register is zero, executes the next instruction in sequence.

Timing: 1 or 3 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Registers altered: P and effective jump address

* Applicable to the Varian 620/f-100 only.

JBNZM Jump and Mark if B Register Not Zero*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										026									
i	Jump Address																		

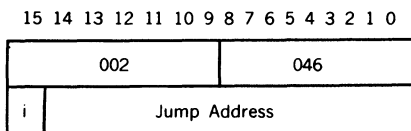
If the B register is not zero, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If the B register is zero, executes the next instruction in sequence.

INSTRUCTION SET

Timing: 1 or 3 cycles
Relative addressing: No
Indirect addressing: Yes
Indexing: No
Registers altered: P and effective jump address

* Applicable to the Varian 620/f-100 only.

JXNZM Jump and Mark if X Register Not Zero



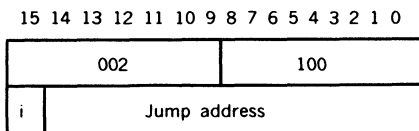
If the X register is not zero, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If the X register is zero, executes the next instruction in sequence.

Timing: 1 or 3 cycles
Relative addressing: No
Indirect addressing: Yes
Indexing: No
Registers altered: P and effective jump address

* Applicable to the Varian 620/f-100 only.

JS1M

Jump and Mark if SENSE Switch 1 Set

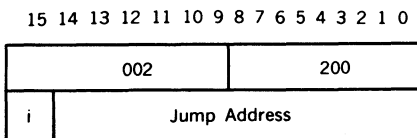


If SENSE switch 1 is set, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If SENSE switch 1 is not set, executes the next instruction in sequence.

Timing:
f-100: 1 or 3 cycles
L-100: 2 or 3 cycles
Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P and effective jump address

JS2M

Jump and Mark if SENSE Switch 2 Set



If SENSE switch 2 is set, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If SENSE switch 2 is not set, executes the next instruction in sequence.

Timing:

f-100: 1 or 3 cycles

L-100: 2 or 3 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P and effective jump address

JS3M**Jump and Mark if
SENSE Switch 3 Set**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										400									
i	Jump Address																		

If SENSE switch 3 is set, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If SENSE switch 3 is not set, executes the next instruction in sequence.

Timing:

f-100: 1 or 3 cycles

L-100: 2 or 3 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Register altered: P and effective jump address

JS1NM**Jump and Mark if
SENSE Switch 1 Not Set***

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										106									
i	Jump Address																		

If SENSE switch 1 is not set, stores the contents of the P register at the jump address, and executes the instruction at jump address plus one. If SENSE switch 1 is set, executes the next instruction in sequence.

Timing:

1 or 3 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Registers altered: P and effective jump address

* Applicable to the Varian 620/f-100 only.

JS2NM**Jump and Mark if
SENSE Switch 2 Not Set***

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										206									
i	Jump Address																		

If SENSE switch 2 is not set, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If SENSE switch 2 is set, executes the next instruction in sequence.

Timing:

1 or 3 cycles

Relative addressing: No

Indirect addressing: Yes

Indexing: No

Registers altered: P and effective jump address

* Applicable to the Varian 620/f-100 only.

INSTRUCTION SET

JS3NM Jump and Mark if SENSE Switch 3 Not set*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										406										
i	Jump Address																			

If SENSE switch 3 is not set, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If SENSE switch is set, executes the next instruction in sequence.

Timing:	1 or 3 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	P and effective jump address

* Applicable to the Varian 620/f-100 only.

JIFM Jump and Mark if Condition(s) Met

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

002										M										
i	Jump Address																			

Analogous to JIF. If all jump conditions are met, jumps to the effective jump address, stores the contents of the P register there, and executes the instruction at the location following the effective jump address. If the jump conditions are not met, executes the next instruction in sequence.

The microcoded jump conditions are established and implemented as described for the JIF instruction in the previous subsection.

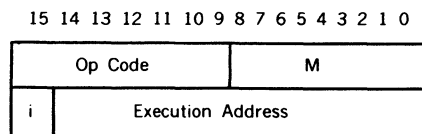
Timing:	
f-100:	1 or 3 cycles
L-100:	2 or 3 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	Those specified

Execution Instructions

This group comprises the instructions that direct the program to a nonsequential address for execution of the instruction located there, and then direct the program back to the main sequence to execute the instruction following the two-word execution instruction.

Mnemonic	Instruction
XEC	Execute unconditionally
XOF	Execute if overflow indicator set
XAP	Execute if A register positive
XAN	Execute if A register negative
XAZ	Execute if A register zero
XBZ	Execute if B register zero
XXZ	Execute if X register zero
XS1	Execute if SENSE switch 1 set
XS2	Execute if SENSE switch 2 set
XS3	Execute if SENSE switch 3 set
XIF	Execute if condition(s) met

These instructions have the following two-word addressing format.



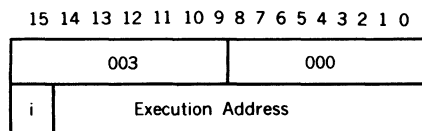
This format comprises a seven-bit operation code (bits 9-15), a nine-bit M field (bits 0-8) in the first word, and an effective execution address in the second.

These instructions have configuration 0 000 011 in bits 9-15.

The M field bits have the same significance as the equivalent jump instruction.

The effective address in the second word is the address of the next instruction to be executed if the execution condition is met. After executing that instruction, the program returns to the main sequence and executes the next instruction in sequence. Note that only one-word instructions that do not specify relative addressing can be contained in the execution address.

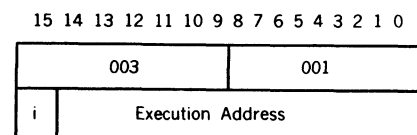
XEC Execute Unconditionally



Executes the instruction at the effective execution address in the second word, and then returns to execute the instruction following XEC.

Timing:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

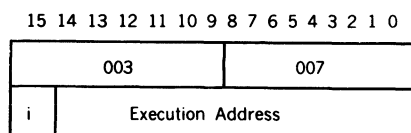
XOF Execute if Overflow Indicator Set



If the overflow indicator is set, executes the instruction at the effective execution address, and then returns to execute the instruction following XOF. Resets the overflow indicator. If the overflow indicator is not set, executes the next instruction in sequence.

Timing:	
f-100:	1 or 2 cycles
L-100:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	OF

XOFN Execute if Overflow Indicator Not Set*



If the overflow indicator is not set, executes the instruction at the effective execution address, and then returns to execute the instruction following XOFN. If the overflow indicator is set, executes the next instruction in sequence. Does not reset OF.

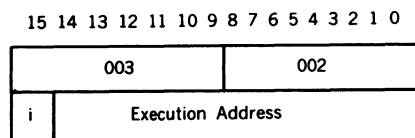
Timing:	1 or 2 cycles
---------	---------------

INSTRUCTION SET

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

* Applicable to the Varian 620/f-100 only.

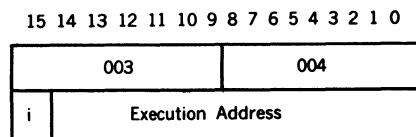
XAP Execute if A Register Positive



If the A register contains a positive value (including zero), executes the instruction at the effective execution address, and then returns to execute the instruction following XAP. If the A register contains a negative value, executes the next instruction in sequence.

Timing:	
f-100:	1 or 2 cycles
L-100:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

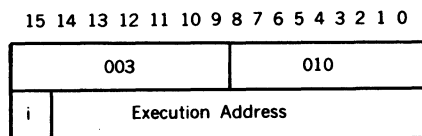
XAN Execute if A Register Negative



If the A register contains a negative value, executes the instruction at the effective execution address, and then returns to

execute the instruction following XAN. If the A register contains a positive value (including zero), executes the next instruction in sequence.

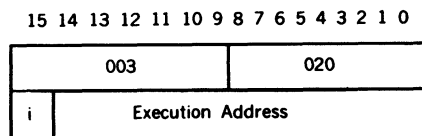
XAZ Execute if A Register Zero



If the A register contains zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the A register does not contain zero, executes the next instruction in sequence.

Timing:	
f-100:	1 or 2 cycles
L-100:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

XBZ Execute if B Register Zero



If the B register contains zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the B register does not contain zero, executes the next instruction in sequence.

Timing:	
f-100:	1 or 2 cycles
L-100:	2 cycles

INSTRUCTION SET

Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

Timing: 1 or 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

XXZ Execute if X Register Zero

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

003										040									
i	Execution Address																		

If the X register contains zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the X register does not contain zero, executes the next instruction in sequence.

Timing:
 f-100: 1 or 2 cycles
 L-100: 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

XANZ Execute if A Register Not Zero*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

003										016									
i	Execution Address																		

If the A register contents are not zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the A register contains zero, executes the next instruction in sequence.

* Applicable to the Varian 620/f-100 only.

XBNZ Execute if B Register Not Zero*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

003										026									
i		Execution Address																	

If the B register contents are not zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the B register contains zero, executes the next instruction in sequence.

Timing: 1 or 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

* Applicable to the the Varian 620/f-100 only.

XXNZ Execute if X Register Not Zero*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

003										046									
i		Execution Address																	

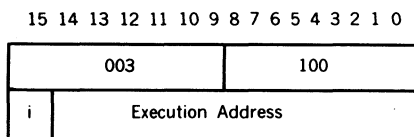
INSTRUCTION SET

If the X register contents are not zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the X register contains zero, executes the next instruction in sequence.

Timing: 1 or 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

* Applicable to the Varian 620/f-100 only.

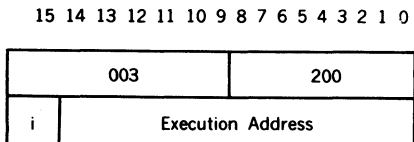
XS1 Execute if SENSE Switch 1 Set



If SENSE switch 1 is set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 1 is not set, executes the next instruction in sequence.

Timing:
 f-100: 1 or 2 cycles
 L-100: 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

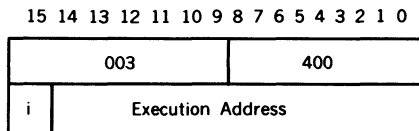
XS2 Execute if Sense Switch 2 Set



If SENSE switch 2 is set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 2 is not set, executes the next instruction in sequence.

Timing:
 f-100: 1 or 2 cycles
 L-100: 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

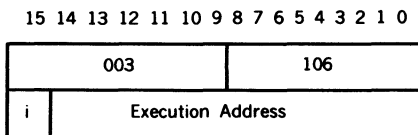
XS3 Execute if SENSE Switch 3 Set



If SENSE switch 3 is set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 2 is not set, executes the next instruction in sequence.

Timing:
 f-100: 1 or 2 cycles
 L-100: 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

XS1N Execute if SENSE Switch 1 Not Set*



INSTRUCTION SET

If SENSE switch 1 is not set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 1 is set, executes the next instruction in sequence.

Timing: 1 or 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

* Applicable to the Varian 620/f-100 only.

XS2N Execute if SENSE Switch 2 Not Set*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

003										206										
i	Execution Address																			

If SENSE switch 2 is not set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 2 is set, executes the next instruction in sequence.

Timing: 1 or 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

* Applicable to the Varian 620/f-100 only.

XS3N Execute if SENSE Switch 3 Not Set*

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

003										406									
i	Execution Address																		

If SENSE switch 3 is not set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 3 is set, executes the next instruction in sequence.

Timing: 1 or 2 cycles
 Relative addressing: No
 Indirect addressing: Yes
 Indexing: No
 Register altered: None

* Applicable to the Varian 620/f-100 only.

XIF Execute if Condition(s) Met

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

003										M									
i		Execution Address																	

Analogous to JIF. If all execution conditions are met, executes the instruction at the effective execution address, and then returns to execute the instruction following XIF. If all execution conditions are not met, executes the next instruction in sequence.

The microcoded execution conditions are established and implemented as described for the JIF instruction.

INSTRUCTION SET

Timing:	
f-100:	1 or 2 cycles
L-100:	2 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

Control Instructions

This group comprises general control instructions.

Mnemonic	Instruction
HLT	Halt
NOP	No operation
ROF	Reset overflow indicator
SOF	Set overflow indicator

These instructions have one-word, nonaddressing formats.

HLT	Halt
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	
00	0 xxx

Stops computation and places the computer in step mode. To restart computation with the next instruction in sequence, press RUN or START on the control panel.

Bits 0-8 can contain any value, e.g., that assigned to specific halt to be displayed on the control panel in the instruction register.

Timing:	1 cycle
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

NOP No Operation

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	
0050	0 0

Waits one cycle. The P register is incremented by one, but the A, B, and X registers and memory are not affected.

Timing:	1 cycle
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

ROF Reset Overflow Indicator

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	
007400	

Resets the overflow indicator (OF).

Timing:	1 cycle
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	OF

SOF Set Overflow Indicator

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	
007401	

Sets the overflow indicator.

Timing:	1 cycle
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	OF

I/O Instructions

This group comprises the instructions for implementing communication between the computer and the peripheral devices that supply and receive data.

Mnemonic Instruction

EXC	External control
SEN	Program sense
CIA	Clear and input to A register
CIB	Clear and input to B register
CIAB	Clear and input to A and B registers
INA	Input to A register
INB	Input to B register
INAB	Input to A and B registers
OAR	Output from A register
OBR	Output from B register
OAB	Output from A and B registers
IME	Input to memory
OME	Output from memory

The format of one-word I/O instructions is:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Op Code						M		A							

where the M field designates the register, line, or function involved, and the A field, a peripheral device address (da).

The formats of two-word I/O instructions are identical with those of analogous operation instructions previously described.

EXC External Control

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0		
100	f	da

Places on the E bus a nine-bit order for the peripheral device to perform function f.

Timing:	
f-100:	2.4-3.8 cycles
L-100:	1 cycle
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

SEN Program Sense

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

101	q	da
i	Jump Address	

Places on the E bus a nine-bit order to sense the status of line q in the peripheral device.

If the computer receives a true response signal, jumps to the instruction at the effective jump address and executes it next. If the response signal is false, executes the next instruction in sequence.

Timing:	
f-100:	2.4-4.6 cycles
L-100:	2.25 cycles
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

CIA Clear and Input to A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

102	5	da
-----	---	----

CIB Clear and Input to B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

102	6	da
-----	---	----

CIAB Clear and Input to A and B Registers

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

102	7	da
-----	---	----

Clears the specified register(s) and inputs a data word from the peripheral device to the specified register(s).

Timing:
f-100: 4.9-6.2 cycles
L-100: 2 cycles

Relative addressing: No
Indirect addressing: No
Indexing: No
Registers altered: Only those specified

INA Input to A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

102	1	da
-----	---	----

INB Input to B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

102	2	da
-----	---	----

INAB Input to A and B Registers

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

102	3	da
-----	---	----

Inclusively ORs a data word from the peripheral device with the contents of the specified register(s), and places the result in the specified register(s).

Timing:
f-100: 4.9-6.2 cycles
L-100: 2 cycles
Relative addressing: No
Indirect addressing: No
Indexing: No
Registers altered: Only those specified

OAR Output From A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

103	1	da
-----	---	----

OBR Output From B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

103	2	da
-----	---	----

OAB Output Inclusive-OR of A and B Registers

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

103	3	da
-----	---	----

Outputs the inclusive-ORed contents of the specified register(s) to the peripheral device.

Timing:
f-100: 4.9-6.2 cycles
L-100: 2 cycles
Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: None

IME Input to Memory

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

102	0	da
0	Memory Address	

INSTRUCTION SET

Inputs a data word from the peripheral device to the cleared memory address in the second word.

Timing:

f-100: 5.9-7.2 cycles

L-100: 3 cycles

Relative addressing: No

Indirect addressing: No

Indexing: No

Register altered: Memory

OME Output From Memory

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

103		0	da
0	Data Address		

Outputs the contents of the memory address in the second word to the peripheral device.

Timing:

f-100: 5.9-7.2 cycles

L-100: 3 cycles

Relative addressing: No

Indirect addressing: No

Indexing: No

Register altered: None

SECTION 17 - VARIAN 620 DAS ASSEMBLERS

The **Varian 620 Assembler Language (DAS)** translates symbolically coded instructions, directives, and data (source program) into their binary machine-language equivalents (object program). DAS allows the programmer to specify instructions, addresses, address modifications, and constants in a manner that is straightforward and meaningful to the computer.

Using DAS, the programmer generates a source program by coding instruction and directive mnemonics rather than numerical values. Memory addresses can be referenced symbolically, thus providing flexibility not attainable with absolute addressing. Constants can be used without prior conversion to binary or octal values. For ease in checkout and program documentation, comments can be added between symbolic source statements, or appended to the statements themselves.

DAS coding reduces machine-language bookkeeping to fully utilize computer capabilities without a corresponding compromise of an increase in the time required for programming.

Three versions of DAS are available:

- a. DAS 4A operates in a minimum-configuration Varian 620-series system comprising the computer, 4K of memory, and an on-line Teletype.

- b. DAS 8A requires a minimum of 8K of memory, and has extended capabilities compared to DAS 4A. Both DAS 4A and DAS 8A can operate with additional system peripherals.
- c. DAS MR operates in expanded computer systems under the control of the Varian 620 Master Operating System (MOS), document number 98 A 9952 091.

DAS processes source programs in two passes. The first pass defines user-designated symbols. The second pass produces an assembly listing and the object program.

Character Set

The DAS character set comprises:

Alphabetical characters

ABCDEFGHIJKLMNOPQRSTUVWXYZ\$

Numerical characters

0123456789

Teletype characters

CR (Carriage return)
LF (line feed)

Special characters

+	(plus sign)
-	(minus sign)
*	(asterisk)
/	(slash)
.	(period)
	(blank)
@	(at sign)
[	(left bracket)
]	(right bracket)
<	(less than)
>	(greater than)
↑	(up arrow)
←	(left arrow)
=	(equal sign)
,	(comma)
'	(prime)
(	(left parenthesis)
)	(right parenthesis)
\	(backslash)
!	(exclamation point)
"	(quotation mark)
#	(pound sign)
%	(percent sign)
+	(ampersand)
:	(colon)
;	(semicolon)
?	(question mark)

Format

DAS source programs are sequences of source statements (records). Each source statement comprises a combination of label, operation, variable, and comment fields, depending on the requirements of the computer instruction or assembler directive, and except in certain cases (described later in this section) generates one computer word.

Label Field

Symbols in the label field identify program points for reference by other parts of the program. They make a program point or

particular numeric value more easily identifiable. The first appearance of a symbol in the label field establishes its identity throughout the remainder of the program. A previously established symbol is referenced by placing it in the variable field of the source statement, where DAS substitutes the previously assigned value from its symbol table.

For DAS 4A and DAS 8A, symbols in the label field comprise one to four alphanumeric characters; for DAS MR there are from one to six such characters. In any case, the first character is alphabetical or the pound sign (#). While only the given number of characters are recognized by DAS, additional characters can be added for programming convenience and/or documentation.

Symbols are usually attached only to those source statements referenced elsewhere in the program, but this is not mandatory.

Operation Field

This source statement field contains mnemonics for computer instructions (section 16) and assembler directives (defined later in this section). An asterisk following the mnemonic specifies **indirect addressing** (section 15). The mnemonics can be redefined with OPSY assembler directives (see below).

Variable Field

The purpose of this field varies according to the requirements of the operation defined by the source statement. The variable field can contain a symbol, a constant, or an expression combining symbols and constants.

DAS Expressions are similar to arithmetic expressions except that parentheses are not used. The variable field can contain the following operators.

+	(addition)
-	(subtraction)
*	(multiplication)
÷	(division)

Arithmetic operations always involve all 16 bits of the computer words, and are performed from left to right, with multiplication and division occurring before addition and subtraction. Thus, $A + B/C * D$ in DAS is equivalent to $A + B(B/C) * D$ in conventional notation.

Coding an asterisk in the first position of the variable field gives access to the then current value of the program location counter. Such an asterisk immediately precedes another operator, and this is the only case in which two adjacent operators are permitted in DAS. The asterisk is translated as the current program location (i.e., $*+1$ means the current program location plus one).

In the following descriptions of DAS constants, *unsigned numbers are considered positive* DAS recognizes decimal and octal integers; floating-point numbers; alpha, address, and indirect address constants; and literals.

A **decimal integer** is a signed or unsigned string of from one to six decimal digits, the first of which cannot be zero (so as not to be confused with octal integers).

Example:

1 29 -3 -9000

An **octal integer** is a signed or unsigned string of from one to seven octal digits, the first of which is zero.

Example:

07 -044 +022745

A **floating-point number** has the form: $\pm \text{integer.fraction} \pm \text{exponent}$, where the right parenthesis, at least one digit, and the decimal point are always present. Other items in the format are optional.

Examples:

0375.64E +7)9.E -2,1E +12
)- 4. +20

An **alpha constant** is a string of characters within primes ('), where, within DAS each character is represented in eight-bit ASCII code. Thus, each 16-bit memory address can hold two characters. Note that blanks are also recognized as characters.

In DAS 4A and DAS 8A, an alpha constant can be a term in an arithmetic expression. However, if more than one word is generated by the constant, only the last word is subject to arithmetic manipulation.

Examples:

'A'*0400 'AB' + 1 'ABCD' + 011

where, in the last example, two words are generated and 011 added to the second word.

An **address constant** is a symbol, number, or expression enclosed in parentheses. It generates a 15-bit direct address (bit 15 = 0). Addressing modes are discussed in section 15.

Examples:

(aaaa + 2) (31) (aaaa)

where aaaa is an address symbol whose value is taken from the symbol table by DAS.

An **indirect address constant** is an *address constant* followed by an asterisk. It generates a 15-bit indirect address (bit 15 = 1).

Examples:

(aaaa + 2)* (3)* (aaaa)*

Literals provide a method for creating and referencing data by expressing the value of the information instead of its address. DAS determines the address and inserts it in the referencing statement and generates a literal table, discarding duplicate values in the table.

A literal is any format of a one-word constant preceded by an equal sign. In a statement requiring more than one literal, they are separated by commas.

Examples:

= 29 = - 044 = (aaaa + 2)*
= 'GO' = 'A'

Comments Field

This field is used for programming notes. An entire source statement can be commentary if an asterisk is coded in the first position. The assembler ignores all comments in the assembly process, but lists them with the program listing output.

Computer Instructions

DAS assemblers recognize the complete instruction sets of all Varian 620-series computers, even when the system on which they operate lacks the hardware for executing a particular instruction. **The programmer, therefore, must have a thorough knowledge of the instructions applicable to his system before attempting to assemble a program.**

Computer instructions, divided by function, are described in detail in section 16. In appendix A, they are listed, arranged alphabetically by mnemonic, with their octal codes, and indexed to the page of this Handbook in which they are discussed. Instruction formats are given in section 14, and section 15 discusses addressing modes.

In this section, all Varian 620-series instructions are divided into five types, according to assembler format requirements.

All Varian 620-series instruction in DAS have the general field format

Label Operation Variable Comments

where the label field is optional and contains a symbol when used; the operation field contains the instruction mnemonic; the variable field contains one, two, or three expressions (separated by commas when there is more than one), and the comments field is optional.

Addressing

If an assembler source statement specifies an address in the first 2,048 words of memory without indirect addressing, the

assembler generates an instruction with direct addressing.

If indexing is specified, the assembler generates an indexed instruction.

Specifying indirect addressing with a data address lower than 512 generates an instruction with indirect addressing and the specified effective memory address.

In all other cases, including indirect addressing with an address higher than 511, the assembler generates an instruction with indirect addressing and the specified effective memory address, stores the address in a table, and inserts the storage address in the referencing instruction. Duplicate values in the table are discarded.

The Varian 620/L-100 permits indirect addressing to an unlimited number of levels. In the Varian 620/f-100, indirect addressing is limited to five levels with one-word instructions and to four levels with two-word instructions.

Instruction Types

Table 17-1 summarizes the characteristics of the five types of computer instructions for DAS use. Instruction mnemonics are given in the applicable type description below and summarized in table 17-2.

Assembler type 1 instructions are:

ADD	LDA	STA
ANA	LDB	STB
DIV	LDX	STX
ERA	MUL	SUB
INR	ORA	

An assembler type 1 instruction occupies one computer word and is memory-addressing.

Indirect addressing is specified by an asterisk after the mnemonic or after a variable field expressed in parentheses.

Examples:

```
LDA* expression
LDA (expression)*
```

Indexing is specified by two expressions in the variable field. The first is the indexing increment and is less than 0512. The second specifies the indexing register: X register = 1, and B register = 2. These instructions cannot be postindexed.

Example:

```
LDA 0300,1
```

loads the A register with the contents of the memory address specified by the sum of the X register contents and 0300. Thus, if the X register contains 0200, the operand for this instruction is in memory address 0500.

Assembler type 2 instructions are:

ADDI	JOFN	STXI
ANAI	JOFMN	SUBI
DIVI	JSS1	XAN
ERAI	JSS2	XANZ
INRI	JSS3	XAP
JAN	JS1M	XAZ
JANM	JS1NM	XBNZ
JANZ	JS2M	XBZ
JANZM	JS2NM	XEC
JAP	JS3M	XOF
JAPM	JS3NM	XOFN
JAZ	JXZ	XS1
JAZM	JXZM	XS1N
JBZ	LDAI	XS2
JBZM	LDBI	XS2N
JMP	LDXI	XS3
JMPM	MULI	XS3N
JO	ORAI	XXNZ
JOFM	STAI	XXZ
	STBI	

Table 17-1. Assembler Instruction Type Characteristics

Parameter	Type 1	Type 2	Type 3	Type 4	Type 5
Words generated	1	2	2	1	2
Memory addressed	Yes	Yes*	Yes	No	Yes
Indirect addressing	Yes	Yes*	Yes	No	Yes
Indexing	Yes	No	No	No	Yes
Variable field expressions	1 or 2	1	2	1	1 to 3
Microcoding	No	No	Yes	Yes	No

* Except for Immediate Instructions

Table 17-2. Summary of Assembler Instruction Types

Type 1	Type 2		Type 3	Type 4		Type 5
ADD	ADDI	JS3NM	BT	AOFA	LASR	ADDE
ANA	ANAI	JXZ	IME	AOFB	LLRL	ANAE
DIV	DIVI	JXZM	JIF	AOFX	LLSR	DIVE
ERA	ERAI	LDAI	JIFM	ASLA	LRLA	ERAE
INR	INRI	LDBI	JMIF	ASLB	LRLB	IJMP
LDA	JAN	LDXI	OME	ASRA	LSRA	INRE
LDB	JANM	MULI	SEN	ASRB	LSRB	JSR
LDX	JANZ	ORAI	XIF	CIA	MERG	LDAE
MUL	JANZM	STAI		CIAB	NOP	LDBE
ORA	JAP	STBI		CIB	OAB	LDXE
STA	JAPM	STXI		COMP	OAR	MULE
STB	JAZ	SUBI		CPA	OBR	ORAE
STX	JAZM	XAN		CPB	ROF	SRE
SUB	JBZ	XANZ		CPX	SEL	STAE
	JBZM	XAP		DAR	SEL2	STBE
	JMP	XAZ		DBR	SOF	STXE
	JMPM	XBNZ		DECR	SOFA	SUBE
	JOE	XBZ		DXR	SOFB	
	JOEM	XEC		EXC	SOFX	
	JOEN	XOF		EXC2	TAB	
	JOENM	XOFN		HLT	TAX	
	JSS1	XS1		IAR	TBA	
	JSS2	XS1N		IBR	TBX	
	JSS3	XS2		INA	TXA	
	JS1M	XS2N		INAB	TXB	
	JS1NM	XS3		INB	TZA	
JS2M	XS3N		INCR	TZB		
JS3M	XXZ		LASL	ZERO		

An assembler type 2 instruction occupies two consecutive computer words and is memory-addressing. The second word is the address of a jump, jump-and-mark, or execution instruction or the operand specified by an immediate instruction.

Indirect addressing is specified as with an assembler type 1 instruction. These instructions cannot be indexed.

Assembler type 3 instructions are:

BT	JIFM	SEN
IME	JMIF	XIF
JIF	OME	

An assembler type 3 instruction occupies two consecutive computer words and is memory-addressing. It differs from an assembler type 2 instruction in that the variable field contains two expressions to implement instruction microcoding as described below.

For the JIF, JIFM, JMIF, and XIF instructions, the first expression specifies the conditions required for the jump, jump-and-mark, or execution. The condition(s) are specified according to the rules given in section 16 and summarized below. As indicated, multiple conditions can be specified by setting additional bits.

Variable Field	Jump/Execute if:
0001	Overflow indicator is set
0002	A register contents are positive
0004	A register contents are negative
0010	A register contents are zero
0020	B register contents are zero

0040	X register contents are zero
0100	SENSE switch 1 is set
0200	SENSE switch 2 is set
0400	SENSE switch 3 is set

Example:

JIF 0222,ALFA

Takes the next instruction from symbolic address ALFA if the A register contains a positive number (0002), the B register contains zero (0020), and SENSE switch is set (0200); i.e., $0002 + 0020 + 0200 = 0222$.

For the SEN instruction, the first expression specifies the device address and the I/O function; for IME and OME, the device address.

For the BT instruction, the first expression specifies the register and bit to be tested (section 16).

Example:

BT 056,ADDR

takes the next instruction from symbolic address ADDR if bit 14 of the A register contents is zero.

Indirect addressing is specified by an asterisk after the mnemonic or after a variable field expression in parentheses as described for the type 1 instructions.

Note: IME and OME cannot specify indirect addressing.

Assembler type 4 instructions are:

AOFA	EXC2	OAR
AOFB	HLT	OB
AOFX	IAR	ROF
ASLA	IBR	SEL
ASLB	INA	SEL2
ASRA	INAB	SOF
ASRB	INB	SOFA
CIA	INCR	SOFB
CIAB	IXR	SOFX
CIB	LASL	TAB
COMP	LASR	TAX
CPA	LLRL	TBA
CPB	LLSR	TBX
CPX	LRLA	TXA
DAR	LRLB	TXB
DBR	LSRA	TZA
DECR	LSRB	TZB
DXR	MERG	TZX
EXC	NOP	ZERO
	OAB	

An assembler type 4 instruction occupies one computer word and does not address memory.

For COMP, DECR, INCR, MERG, and ZERO and the register transfer/modification instructions, the assembler generates an instruction as specified by the value in the variable field. This value is determined by coding the summed octal value of the possible binary configurations described for these instructions in section 16.

Example:

COMPL 0235

unconditionally takes the inclusive-OR and complements (0200) the contents of the A (0010) and B (0020) registers, and places the result in the A (0001) and X (0004) registers. Note that if bit 8 were one in the above example (0635) the instruction is executed only if the overflow indicator is set.

For EXC, SEL, and SEL2, the expression specifies the I/O function and the device address; for the remainder of the I/O instructions in this group, the device address only (the I/O function being specified by the mnemonic).

Example:

CIB 030

clears the B register and loads it from peripheral specified by the device address 030 (standard device addresses are given in section 7).

Assembler type 5 instructions are:

ADDE	INRE	SRE
ANAE	LDAE	STAE
DIVE	LDBE	STBE
IJMP	LDXE	STXE
ERA	MULE	SUB
JSR	ORAE	

An assembler type 5 instruction occupies two consecutive computer words and is memory-addressing.

Indirect addressing is specified by an asterisk after the mnemonic or after a variable field expression in parentheses as described for the type 1 instructions.

Indexing the 620/L-series instructions is specified as described for the type 1 instructions.

Preindexing the 620/f-series instructions is specified as described for the type 1 instructions. Note that IJMP and SRE cannot be preindexed.

Postindexing the 620/f-series instructions is specified by three expressions in the variable field. The first expression is the data address, the second specifies the

indexing register (X register = 1, and B register = 2), and the third is logically ORed with the instruction word to set bit 7 (which specifies postindexing). The assembler does not check the validity of the third expression, thus one should always use the value 0200.

Example:

```
LDAE    ADDR,2,0200
```

loads the A register extended and postindexed with the B register.

JSR can be neither preindexed nor postindexed.

For SRE, the first expression in the variable field is the data address, the second specifies the type of addressing (1 = indexed with X, 2 = indexed with B, and 7 = direct /indirect), and the third is logically ORed with the instruction word to control bits 3-5 to specify the register to be compared (010 = A register, 020 = B register, and 040 = X register). Note that indirect addressing is specified by an asterisk following the instruction mnemonic.

Examples:

```
SRE    ADDR,7,020
```

compares the contents of the B register with the directly addressed word at ADDR, and, if equal, skips the next two locations.

```
SRE*   ADDR,1,010
```

compares the contents of the A register with the word at ADDR, using indirect addressing and postindexing with the X register.

Assembler Directives

Directives are instructions to the assembler. They are divided into the following functional groups:

- Symbol definition
- Instruction definition
- Location counter control
- Data definition
- Memory reservation
- Conditional assembly
- Assembler control
- Subroutine control
- List and punch control
- DAS 8A interface to stand-alone FORTRAN
- Program linkage
- MOS I/O control
- Macro definition

Assembler directives have the same general format as the computer instructions. In the following descriptions of the individual directives, the field format

label	operation	variable
-------	-----------	----------

is used, with the optional comment field being understood to follow the variable field when used. In cases where the variable field contains more than one item or expression, these are always separated by commas. Mandatory elements of the directive are in **bold type**, and optional items, in *italic type*.

Table 17-3 summarizes the assembler directives (arranged by function) and indicates those recognized by each DAS assembler.

Table 17-3. Directives Recognized by DAS Assemblers

Function	Directive	DAS 4A	DAS 8A	DAS MR
Symbol definition	EQU	Yes	Yes	Yes
	SET	Yes	Yes	Yes
	MAX	No	Yes	No
	MIN	No	Yes	No
Instruction definition	OPSY	No	Yes	Yes
Location counter control	ORG	Yes	Yes	Yes
	LOC	Yes	Yes	Yes
	BEGI	Yes	Yes	Yes
	USE	Yes	Yes	No
Data definition	DATA	Yes	Yes	Yes
	PZE	Yes	Yes	Yes
	MZE	Yes	Yes	Yes
	FORM	No	Yes	No
Memory reservation	BSS	Yes	Yes	Yes
	BES	Yes	Yes	Yes
	DUP	No	Yes	Yes
Conditional assembly	IFT	No	Yes	Yes
	IFF	No	Yes	Yes
	GOTO	No	Yes	Yes
	CONT	No	Yes	Yes
	NULL	No	Yes	Yes
Assembler control	MORE	No	Yes	No
	END	Yes	Yes	Yes
Subroutine control	ENTR	Yes	Yes	Yes
	RETU*	Yes	Yes	Yes
	CALL	Yes	Yes	Yes
List and punch control	LIST	No	Yes	No
	NLIS	No	Yes	No
	SMRY	No	Yes	Yes
	DETL	No	Yes	Yes
	PUNC	No	Yes	No
	NPUN	No	Yes	No
	SPAC	No	Yes	No
	EJEC	No	Yes	Yes
	READ	No	Yes	No

Table 17-3. Directives Recognized by DAS Assemblers (continued)

Function	Directive	DAS 4A	DAS 8A	DAS MR
Interface to FORTRAN	FORT	No	Yes	No
Program linkage	NAME	No	Yes	Yes
	EXT	No	Yes	Yes
	COMM	No	Yes	Yes
MOS I/O control	Applicable to DAS MR only refer to Varian 620 MOS manual, document number 98 A 9952 091.			
Macro definition	MAC	No	No	Yes
	EMAC	No	No	Yes

Symbol Definition Directives

EQU (DAS 4A, DAS 8A, DAS MR)

These directives assign arbitrary values to symbols in the symbol table. This table is a list of symbols appearing in the source program. For each symbol in the table, there is a corresponding value, usually an address in memory. DAS symbol table capacities are summarized in table 17-4.

This directive has the format

symbol EQU expression

It places the **symbol** in the assembler's symbol table and assigns it the value of the **expression**. If the symbol has already been entered in the symbol table, DAS

Table 17-4. DAS Symbol Table Capacities

Assembler	4K Memory	8K Memory	>8K Memory
DAS 4A	150	1,450	1,450 + n (1,300)
DAS 8A		440	440 + n (800)
DAS MR		20	20 + n (800)

where n = number of 4K memory increments above 8K.

VARIAN 620 DAS ASSEMBLERS

outputs error message ***DD** (described later in this section), and the expression replaces the value in the symbol table. If a symbol is used as the variable field expression, it must have been previously defined. **The label field symbol is mandatory.**

SET (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol SET expression

It is the same as EQU, except that there is no error message output if the symbol has already been entered in the symbol table.

MAX (DAS 8A)

This directive has the format

symbol MAX expression,expression(s)

It assigns the largest algebraic value found among the **expressions** to the **symbol**. If a symbol is used as a variable field expression, it must have been previously defined. The label field symbol is mandatory. Use SET to redefine the symbol.

MIN (DAS 8A)

This directive has the format

symbol MIN expression,expression(s)

It is the same as MAX, except that the symbol is assigned the smallest algebraic value found among the **expressions**.

Instruction Definition Directive

This directive redefines a standard instruction mnemonic.

OPSY (DAS 8A, DAS MR)

This directive has the format

symbol OPSY mnemonic

It makes the symbol a mnemonic with the same definition as the variable field mnemonic.

Example:

```
CLA  OPSY LDA
CLA  BETA
```

Location Counter Control Directives

These directives control the program location counter(s), which control memory area assignments and always point to the next available word.

DAS 8A has several location counters and directives to modify or preset their values. Table 17-5 lists the five standard DAS 8A location counter symbols and their uses. They need not be created by the user. However, up to eight other location counters can be created, thus providing complex relocatable and overlay programs within a single assembly. Relocatability rules are given later in this section.

There are no user-created location counters at the beginning of an assembly. The assembler uses three location counters for program location assignment. Thus, IAOR (indirect pointer assignments) and LTOR (literal assignments) are always in use, as is a third counter used to assign locations to generated instructions and data. The blank location counter performs this task until the USE directive specifies another counter.

In a straightforward program using only one location counter, the ORG and LOC directives completely control the counter.

ORG (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol **ORG** *expression*

It sets the location counter currently in use to the value of the **expression**. If a symbol is present in the label field, it is also set to the value of the expression.

Any symbol used as the variable field expression must have been previously defined.

LOC (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol **LOC** *expression*

It is used to keep a block of data or instructions undisturbed by the assembly process by causing the data and instructions following the LOC address to be moved to the LOC address by the object program before execution. Data or instructions following LOC are generated as if an ORG directive had changed the current location counter value. However, this value is not actually changed.

Any symbol used as a variable field expression must have been previously defined. LOC cannot be used in a relocatable program.

Table 17-5. Standard DAS 8A Location Counters

Counter	Initial Value	Description
COMN	002000	Controls assignment of memory within an interface area common to two or more programs
IAOR	000100	Controls assignment of memory to indirect pointers
LTOR	001000	Controls assignment of memory to literals
SYOR	000000	Controls assignment of memory to all system parameters
blank	004000 000000 for 4A	Used initially and normally by the assembler for memory assignments until/ unless overridden by the USE directive

VARIAN 620 DAS ASSEMBLERS

BEGI (DAS 4A, DAS 8A)

This directive has the format

symbol **BEGI** *expression*

It creates a new location counter, or redefines the value of any location counter before the counter has been used. BEGI gives the new or redefined location counter the value of the **expression**, but has no effect on the current location counter.

BEGI cannot redefine the value of any location counter that has been used for location assignment.

Any symbol used as a variable field expression must have been previously defined.

USE (DAS 8A)

This directive has the format

blank **USE** *xxxx*

where **xxxx** is a blank, COMN, SYOR, or a user-created location counter label.

The USE directive uses location counter **xxxx** to assign locations to data and instructions (except literals and indirect pointers).

If **xxxx** is PREV, the previously used location counter is recalled with the restriction that only the last-used counter can be so recalled.

Data Definition Directives

These directives control the sign and assignment of data words. In the descrip-

tions, **item** refers to a data item, which can be an expression or a direct or indirect address.

DATA (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol **DATA** *item,item(s)*

It generates data words with the values specified by the **items** in the variable field. DATA assigns the *symbol*, if used, to the memory address of the first generated word. In the absence of a symbol, an unlabeled block of data is generated.

When a single alpha constant is used in the variable, DAS 4A and DAS MR left-justify it in the field and fill the remaining positions with blanks, and DAS 8A right-justifies it, filling the remaining positions with zeros.

PZE (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol **PZE** *item,item(s)*

It is similar to DATA except that the sign bit of the generated data word is always zero (positive).

MZE (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol **MZE** *item,item(s)*

It is similar to DATA except that the sign bit of the generated data word is always one (negative).

FORM (DAS 8A)

This directive has the format

symbol **FORM** *term,term(s)*

where the **terms** are absolute terms or expressions.

FORM specifies the format of a bit configuration of a data word. The *symbol*, if used, is the name of the format. The **terms** specify the length in bits of each field in the generated data word, where the sum of their values is from one to the number of bits in the computer word.

FORM is ignored if there are any errors in the variable field, except that an error is flagged when a **term** cannot be represented in the number of bits specified when FORM is applied (by placing its name in the operation field of a symbolic source statement) to another statment. FORM can be redefined.

Example:

BYTE	FORM	8,8
BCD	FORM	4,4,4,4
PTAB	FORM	1,2,3,4

would, given the FORM definition

ABC	FORM	6,2,8
-----	------	-------

and the FORM reference

ABC	FORM	2*3,1,'A'
-----	------	-----------

generate the binary data word

0	001	100	111	000	001
---	-----	-----	-----	-----	-----

Memory Reservation Directives

These directives control the reservation of memory addresses and areas.

BSS (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol **BSS** *expression*

It reserves a block of memory addresses by increasing the value of the current location counter the amount indicated by the **expression**. The *symbol*, if used, is assigned the value of the counter prior to such an increase, thus referencing the starting address of the reserved block.

The location counter always points to the next available word.

If the variable field expression value is zero, the *symbol* is assigned the next available address.

BES (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol **BES** *expression*

It is similar to BSS, except that if there is a symbol it is assigned to the address one less than the incremented location counter.

DUP (DAS 8A, DAS MR)

This directive has the formats

blank	DUP	n
blank	DUP	n,m

It duplicates source statements following its use. The first format duplicates the next source statement the number of times specified by **n**. The second format duplicates the next source statement (the number of which is specified by **m**) the number of times specified by **n**, where $m, \leq 3$ and $n \leq 32,767$. If **n** or **m** is zero, it is treated as if it were a one.

Conditional Assembly Directives

These directives assemble portions of the program according to the conditions specified in the variable fields.

IFT (DAS 8A, DAS MR)

This directive has the format

blank IFT expression,expression(s)

It assembles the next symbolic source statement only if the first **expression** is less than the second, and the second is less than or equal to the third.

Examples:

IFT a
for $a \neq 0$.

IFT a,,b
for $a \neq b$.

IFT a,b,b
for $a < b$.

IFT 0,a,b
for $0 < a \leq b$.

IFF (DAS 8A, DAS MR)

This directive has the format

blank IFF expression,expression(s)

It is similar to IFT (IFT = true), except that IFF (IFF = false) is the logical complement of IFT.

Examples:

IFF a
for $a = 0$.

IFF a,,b
for $a = b$.

IFF a,b,b
for $a \geq b$.

IFF 0,a,b
for $0 \geq a > b$.

GOTO (DAS 8A, DAS MR)

This directive has the formats

blank GOTO symbol
blank GOTO symbol,
blank GOTO integer
blank GOTO integer,

It skips more than one instruction and usually follows an IFF or IFT directive. All source statements between the GOTO and the statement containing the **symbol** in its label field are skipped, and the instruction so labeled executed next. GOTO cannot return to an earlier point in the program.

If the first and third GOTO formats are used, the skipped instructions are listed. If the second and fourth formats (containing a comma after the variable field element) are used, they are not listed. This listing can also be suppressed by a SMRY directive.

CONT (DAS 8A, DAS MR)

This directive has the format

symbol CONT blank

It provides a target for a previous GOTO directive. The **symbol** is not entered in the assembler's symbol table.

NULL (DAS 8A, DAS MR)

This directive has the format

symbol NULL blank

It provides a target for a previous GOTO directive with the **symbol** entered in the symbol table. NULL has the same effect as a BSS directive with a blank variable field.

Assembler Control Directives

These directives signal the end or continuance of an assembly.

MORE (DAS 8A)

This directive has the format

blank MORE blank

It halts the assembly process to allow additional source statements to be put in the input device. Assembly resumes when the RUN or START switch on the computer

control panel is pressed. MORE is never listed.

END (DAS 4A, DAS 8A, DAS MR)

This directive has the format

blank END expression

It is the last source statement in the program. The *expression* is the execution address of the program after it has been loaded into the computer. A blank in the variable field yields an execution address of 000000.

Subroutine Control Directives

These directives create closed subroutines and control their use.

ENTER (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol ENTR blank

where the **symbol** is the name of the subroutine called. ENTR generates a linkage word of zero in the object program.

RETU* (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol RETU* expression

It returns from a closed subroutine, generating an unconditional jump to the address indicated by the value of the expression.

CALL (DAS 4A, DAS 8A, DAS MR)

This directive has the format

symbol CALL name,parameter,error

where

name	is the symbolic name of a subroutine
parameter	is an optional list of parameters comprising valid data items
error	is an optional list of error returns comprising valid data items

CALL causes the program to jump to the closed subroutine specified by **name**. Where a *symbol* is used in the label field, it is entered in the symbol table and assigned the value of the current location counter.

Example:

CALL FUNC,X,Y + 1,(ERR),(GOOF)*

produces a machine code identical to that obtained with

```

JMPM        FUNC
.
.
.
DATA        X,Y, + 1,(ERR),GOOF)*
.
.
.
    
```

List and Punch Control Directives

These directives, which are operative only during the second pass of the assembler (that producing the object program and listings), control listing and punching during program assembly.

List (DAS 8A)

This directive has the format

blank LIST blank

It causes the assembler to produce a program listing. The assembler normally outputs a list of the source statements. The LIST directive is used to bring the assembler back to this condition when any of the following directives change the listing status.

NLIS (DAS 8A)

This directive has the format

blank NLIS blank

It suppresses further listing of the program.

SMRY (DAS 8A, DAS MR)

This directive has the format

blank SMRY blank

It suppresses the listing of source statements that have been skipped under control of the conditional assembly directives.

DETL (DAS 8A, DAS MR)

This directive has the format

blank DETL blank

It removes the effect of SMRY, i.e., causes listing of all source statements, including those skipped by conditional assembly directives.

PUNC (DAS 8A)

This directive has the format

blank PUNC blank

It causes the assembler to produce a paper tape punched with the object program. The assembler normally outputs such a tape. PUNC returns the assembler to this condition when the following directive changes the punching status:

NPUN (DAS 8A)

This directive has the format

blank NPUN blank

It suppresses further production of paper tape punched with the object program.

SPAC (DAS 8A, DAS MR)

This directive has the format

blank SPAC blank

It causes the listing device to skip a line. SPAC is not listed.

EJEC (DAS 8A, DAS MR)

This directive has the format

blank EJEC blank

It causes the listing device to move to the next top of form. EJEC is not listed.

READ (DAS 8A)

This directive has the format

blank READ *number,parameter*

where

<i>number</i>	is the number of characters (20 to 80) from each source statement to be processed by the assembler
<i>parameter</i>	is either 26 or 29 to indicate use of IBM 026 or 029 keypunch codes, respectively

Normally, the assembler processes 80 characters per statement with 026 keypunch codes. If *number* is outside the range 20 to 80, the assembler resets the number of characters to 80 and outputs error message *SZ. If *parameter* is not 26 or 29, assembly stops with the A, B, X, and instruction registers equal to 000026. To continue assembly, correct the card, reinsert it in the reader, and press RUN or START on the computer control panel.

If *number* is used without *parameter*, there is no comma. If *parameter* is used without *number*, it is preceded by a comma.

Unless there is an *SZ error message, the SMRY directive suppresses the listing of READ during the second pass of the assembly process.

Examples:

READ 80,29	Reads 80 columns of 029 codes
READ 72,26	Reads 72 columns of 026 codes
READ ,29	No change in columns, but reads 029 codes
READ 80	Reads 80 columns, but does not change the codes

Interface to FORTRAN Directive

FORT (DAS 8A)

This directive has the format

blank FORT blank

Except for comments, this is the first line of an assembly whose output is to be compatible with the stand-alone FORTRAN relocatable loader.

In such an assembly, all two-word instructions are legal, but one-word memory-addressing instructions are limited to relative forward addressing. An expression containing symbols defined with COMN or EXT directives must be the second word of two-word instructions, or part of a DATA, PZE, or MZE directive. Literals are forbidden. Immediate instructions are recommended. The symbols and expressions are subject to the mode restrictions indicated for FORTRAN-compatible output, which are given in this section following the assembler directive descriptions.

Program Linkage Directives

These directives establish and control links among programs that have been assem-

bled separately but are to be loaded and executed together.

NAME (DAS 8A, DAS MR)

This directive has the format

blank NAME symbol,...,symbol

It establishes linkage definition points among separately assembled programs. Each **symbol(s)** can then be referenced by other programs. Each **symbol** also appears in the label field of a symbolic source statement in the body of the program. Undefined NAME symbols cause error messages to be output.

Under DAS 8A, NAME statements immediately follow the FORT directive.

Examples:

NAME	A
NAME	A,B
NAME	EX,WHY,ZEE

EXT (DAS 8A, DAS MR)

This directive has the format

symbol EXT symbol,...,symbol

In linking separately assembled programs, it declares each **symbol** not defined within the current program. Each **symbol**, in both the label and variable field, is output to the relocatable loader with the address of the last reference to the symbol.

If a symbol is not defined within the current program and not declared in an EXT directive, it is considered undefined and causes an error message output. If a symbol is declared in EXT but not referenced within the current program, it is

output to the loader for loading, but no linkage to this program is established. If a symbol is both defined in the program and declared to be external, the EXT declaration is ignored.

Examples:

	EXT	AY
BEG	EXT	BE,SEE
	EXT	DEE,EE,FF,GEE

COMN (DAS 8A, DAS MR)

This directive has the format

symbol **COMN** *item*

where *item* is an absolute item or expression.

COMN defines an area in blank common for use at execution time. This allows an assembler program to reference the same blank common area as a FORTRAN program. The common area is cumulative for each use of COMN, i.e., the first COMN defines the base area of the blank common, the second COMN defines an area to be added to the already established base, etc.

Examples:

AAA	COMN	3
	COMN	6*2
BBB	COMN	9

MOS I/O Control Directives

DAS MR accepts the MOS control directives listed below and explained in the Varian 620 Master Operating System Manual (document number 98 A 9952 091).

Directive

Description

RBIN	Read binary record
RALF	Read alphanumeric record
RBCD	Read binary-coded decimal (BCD) record
WBIN	Write binary record
WALF	Write alphanumeric record
WBCD	Write BCD record
WEOF	Write end of file
REW	Rewind
SKFF	Skip files forward
SKFR	Skip files reverse
SKRF	Skip records forward
SKRR	Skip records reverse
FUNC	Function
STAT	Status
ION	I/O driver reference number

Macro Definition Directives

These directives begin and end macro definitions. The macro is the assembly equivalent of the execution subroutine. It is defined once and can then be called from the program. The macro is an algorithmic statement of a process that can vary according to the arguments supplied. It is assembled with the resultant data inserted into the program at each point of reference, whereas the subroutine executed during execution time appears but once in a program. Its definition comprises the statements between MAC and EMAC.

MAC (DAS MR)

This directive has the format

symbol **MAC** *blank*

It introduces a macro definition. The *symbol* is the name of the macro.

EMAC (DAS MR)

This directive has the format

blank EMAC blank

It terminates the definition of a macro.

A macro is called by the appearance of its name in the operation field of a symbolic source statement. The variable field of this statement contains expression(s) P(1), P(2),...P(n), then processed with the values in the table being substituted for the respective values of the expressions in the source statement variable field. For example, if the variable field of the symbolic source statement contains

2,B,9 + 8, = 63

then within the generated macro P(1) = 2, P(2) is the value of B, P(3) = 021, and P(4) is the address of the value 63. All terms and expressions within the macro-referencing symbolic source statement parameter list are evaluated prior to calling the macro.

If the label field of such a source statement contains a symbol, the symbol is assigned the value and relocatability of the location counter at the time the macro is called but before data generation.

A macro definition can contain references to machine instruction mnemonics or to assembler directives other than DUP. Macros can be nested within macros to a depth limited only by the available memory at assembly time.

Example: Define the macro.

```
SBR MAC
SEN 0200 P(1),* + 3
JMP *- 2
EMAC
```

Call the macro.

```
SBR 031
```

Expand the macro.

```
SEN 0231,* + 3
JMP *- 2
```

P(0) can also be accessed by a normal call. P(0) is the first entry in the table formed by the assembler and contains the number of entries in that table. The following example shows the output listing obtained by calling P(0):

		1	A	MAC	
		2		DATA	P (0)
		3		EMAC	
000001	000000A	4		A	
000002	000001A	5		A	1
000003	000002A	6		A	1, 2
000004	000003A	7		A	1, 2, 3
000005	000004A	8		A	1, 2, 3, 4
000006	000005A	9		A	1, 2, 3, 4, 5
		10		END	

Symbol And Expression Modes

Each symbol or expression has one of the following modes assigned by the assembler:

- External (E)
- Common (C)
- Relative (R)
- Absolute (A)

The mode of an expression is determined by the mode of the symbols in the expression.

The mode of a symbol is determined by the following rules:

- If the symbol is in an EXT directive, the mode is E.
- If the symbol is defined by a COMM directive, the mode is C.
- If the symbol is a symbol in a program, or if * is the current location counter value, the mode is R.
- If the symbol is a number (numerical constant), the mode is A.
- If the symbol is defined by an EQU, SET, or similar directive, the mode of the symbol is that of the variable field expression in the directive.

The mode of an expression is determined by the following rules:

- If the expression contains any mode E or C symbol, the expression is mode E.

- If the expression contains only mode A symbols, the expression is mode A.
- If the expression contains mode and R symbols, the mode of the expression is R if there is an odd number of mode R symbols. Otherwise, the mode of the expression is A.

The following restrictions apply only to DAS MR and to FORTRAN-compatible output assembly with DAS 8A.

- No expression can contain symbols of both modes E and C.
- A mode E expression comprises a single mode E symbol.
- No mode E, C, or R expression can multiply or divide a mode E or C symbol.
- No expression can add or subtract a mode C and a mode R symbol, or a mode E and a mode R symbol.
- No expression can add two or more mode E, C, or R symbols.
- A mode A symbol can be added to or subtracted from a mode C or R symbol.

Figure 17-1 illustrates the above rules.

Relocatability Rules

A relocatable program (DAS 8A, DAS MR) is one that has been assembled with its instruction and directive locations assigned in such a manner that it can be loaded and executed anywhere in memory. When such a program is loaded, the beginning memory address is specified,

EEEE	EXT		Defines mode E
CCCC	COMM	6	Defines mode C
RTN	ENTR		Defines a symbol (RTN) as a mode R
TBL	BSS	50	TBL is mode R
ABL	BSS	'A' + 5	ABL is mode R
LENG	EQU	*- TBL	LENG is mode A (defines area length)
	CALL	EEEE,TBL,LENG	
	LDA	* + 6	Legal, one-word relative forward
	LDA	CCCC + 6	Illegal, one-word not R or A
	LDXI	CCCC + 6	Legal, two-word instruction
	LDA	0,1	Legal, loads CCCC + 6 in A register
	.		
	.		
	.		
	DATA	EEEE + 4	Illegal, value not zero
	DATA	CCCC + 4	Legal
	DATA	CCCC + LENG	Legal
	DATA	TBL + LENG 5	Legal, mode is R

Figure 17-1. Manipulation of Expression and Symbol Modes

and a value (known as the relocation bias) is added to the addresses of subsequent relocatable instructions. The programs are usually assembled with a zero relocation bias on the first instruction.

The location counter contains the (relative) address of the instruction or directive currently being executed. The location counter is absolute when it contains the actual address of the instruction, and relocatable when it contains the relative

address (the current address of the start of the program).

Symbols can be absolute or relocatable. Expressions, since they contain symbols, can be absolute or relocatable. Constants are always absolute.

The following shows, for each arithmetic operation, whether the result is absolute (abso), relocatable (relo), or illegal.

	A = abso B = abso	A = abso B = relo	A = relo B = abso	A = relo B = relo
A + B	abso	relo	relo	illegal
A - B	abso	illegal	relo	abso
A * B	abso	illegal	illegal	illegal
A / B	abso	illegal	illegal	illegal

The relocatable loader can load a program in any area of memory and modify the addresses as it loads so that the resulting program executes correctly. Programs can contain absolute addresses, relocatable addresses, or both. At the beginning of each instruction or data word generated by the assembler, it can be set by the ORG directive. On encountering an ORG directive, the assembler makes the location counter absolute if the corresponding expression is absolute, or relocatable if the corresponding expressions is relocatable.

If a symbol is equated to the location counter, it is relocatable if the location counter is relocatable. Otherwise, the symbol is absolute.

Assembler Input Media

Punched Card Format

Punched cards used as input to the DAS assemblers contain four fields corresponding to the instruction and directive fields:

- a. The **label field** is in columns 1 through 6. Its use is governed by the requirements of the instruction or directive.
- b. The **operation field** is in columns 8 through 14. It contains the instruction or directive mnemonic. Indirect addressing is specified by an asterisk following the mnemonic.
- c. The **variable field** begins in column 16 and ends with the first blank that is not part of a character string. Its use depends on the instruction or directive. If two or more subfields are present, they are separated by commas.

- d. The **comment field** fills the remainder of the card. If the variable field is blank, the comment field begins in column 17.

An asterisk in column 1 indicates that the entire card contains a comment.

Paper Tape Format

Paper tape used as input to the DAS assemblers contains source statements of up to 80 characters each (not including the carriage return and line feed characters). Each punched statement contains four fields corresponding to the instruction and directive fields. The label, operation, and variable fields are separated by commas, and the comment field starts after the first variable field blank that is not part of a character string. Each statement is terminated by a carriage return (CR) followed by a line feed (LF).

Note that columns 7 and 15 are always unpunched (blank).

- a. **Label field** use is governed by the requirements of the instruction or directive. It is terminated with a comma. If this field is not used, a comma appears as the first character of the source statement.
- b. The **operation field** contains the instruction or directive mnemonic. An asterisk following the mnemonic specifies indirect addressing. This field begins immediately following the label field terminator and is terminated by a comma.
- c. The **variable field** can be blank, or contain one or more subfields separated by commas. Subfields can be voided by using adjacent commas. This field is terminated by a blank

that is not part of a character string, or with a CR or LF.

- d. The **comment field** fills the remainder of the statement (from the terminating blank of the variable field to the next CR or LF).

If the first nonblank character of a source statement is an asterisk, the entire statement is a comment.

Assembler Output Listing

DAS produces a source/object listing of the assembled program, as well as a paper tape containing the object program in reloadable format.

The listing can be obtained in whole or in part as the program is being assembled.

The source (symbolic) program and the object (absolute) program are listed side by side on the listing device. This device is either a Teletype or a line printer.

The listing is output according to the specifications given by the list and punch control directives in the assembly (DAS 8A, DAS MR).

Error analysis during assembly causes the error messages described below to be output on the line following the point of detection.

The following example illustrates the format of the output listing. A line count appears only on DAS MR listings. The addressing modes are: FORTRAN common reference = C, externally defined = E, indirect pointer = I, and absolute or relative = R.

Address	Code	Mode	Line Count	Symbolic Source Statement
014000				ORG 014000
014000	000000			ABS ENTR
014001	001002			JAP* ABS
014002	114000	R		
014003	005211			CPA
014004	001000			JMP* ABS
014005	114000	R		
	000000			END

Error Messages

The assembler checks source statement syntax during both pass 1 and 2. Detectable errors are listed during pass 1. During pass 2, the following information is listed:

- a. Error code
- b. Location counter value
- c. Object code when the instruction is assembled

This information is suppressed by NLIS directives and list-suppression commas in GOTO directives.

The error message appears in the listing line following the statement found to be in error. Each line can hold up to four error messages.

Table 17-6 lists the DAS error codes and their meanings.

Table 17-6. DAS Error Codes

Code	Meaning
*AD	Error in an address expression
*DC	Decimal character in an octal constant
*DD	Illegal redefinition of a symbol or the location counter
*E	Incorrectly formed statement
*EX	Illegally constructed expression
*FA	Floating-point number contains a format error
*IL	First nonblank character of a source statement is invalid (the statement is not processed)
*NR	No memory space available for additional entries in assembler tables
*NS	No symbol in the label field of a SET, EQU, MAC, or FORM directive or no symbol in the label or variable field of an OPSY directive, or no symbol in the variable field of a NAME directive
*OP	Undefined operation field (two No Operation (NOP) instructions are generated in the object program; the remainder of the statement is not processed), or illegal nesting of DUP or MAC directives
*QQ	Illegal use of prime (')
*R	Relocatable item where an absolute item should be defined
*SE	Synchronization error: symbol value in pass 2 is different from that found in pass 1
*SY	Undefined symbol in an expression
*SZ	Expression value too large for a subfield, or a DUP directive specifies that more than three statements are to be assembled
*TF	Undefined or illegal indexing specification
*UC	Undefined character in an arithmetic expression

Table 17-6. DAS Error Codes (continued)

Code	Meaning
*UD	Undefined symbol in the variable field of a USE directive
*XR	Address out of a range for an indexing specification
* =	Illegal use of a literal

Operating The Assembler

DAS 4A Operations

Load the assembler program supplied by Varian into memory using the binary load/dump program (BLD II, section 18.) Execute it by entering a positive, nonzero value in the A register during loading, or by clearing all registers and pressing (SYSTEM) RESET and RUN or START.

During execution, the program first determines the amount of memory required. It then stores in address 000003 a value one less than the lower limit of BLD II. This is the highest address that the assembler can use without destroying part of BLD II.

DAS 4A comprises two sections: The I/O section allows the specification of I/O devices for assembler input and output. The second section is the assembler itself.

I/O Section Definitions

The I/O section of DAS 4A, using the Teletype printer, makes three requests for definitions of I/O devices:

ENTER DEVICE NAME FOR xx

where xx is one of the I/O function names: SI (source input,) LO (list output), or BO (binary output), respectively.

Respond to each request in turn by typing, on the Teletype keyboard, the name of the desired device, followed by a carriage return (CR). Table 17-7 lists the acceptable device names in response to each request. If the default assignment is desired, merely press CR.

If an incorrect device name is typed, the message

DEVICE NAME NOT VALID

is output and the request repeated.

To terminate the output of any line to the Teletype, press RUBOUT. This error correction feature can be used any time during I/O device specification.

When I/O assignments are complete, the I/O section uses BLD II to load the assembler section into memory.

To restart the I/O section before the assembler section is loaded, press STEP, clear all registers, and press (SYSTEM) RESET and RUN or START.

Table 17-7. Acceptable I/O Devices

Assembly Function	Device	Default Assignment
SI (source input)	Teletype paper tape reader: TR Teletype keyboard: TY High-speed paper tape reader: PR Card reader (model 620-22, -23, or -25): CR Card reader (model 620A, micro- VersLogic): CR1	TR
LO (list output)	Teletype printer: TY Line printer (model 620-76): LP Line printer (model 620-75): LP1 Line printer (model 620-77): LP2	TY
BO (binary output)	Teletype paper tape punch: TP High-speed paper tape punch: PP Card punch (model 620-27): CP1	TP

Assembler Section Definitions

When BLD II relinquishes control to the assembler section, the computer halts with 000001 in the program counter (P register). For an assembler pass 1, set SENSE switch 1; for pass 2, reset SENSE switch 1 and set SENSE switches 2 and 3.

If pass 1 is selected, ready the SI device with the source input media and press RUN or START.

For pass 2, ready the SI device with the source input media, ready the BO and LO devices, and press RUN or START.

The END directive terminates both passes 1 and 2. Pass 1 terminates with 000001 in the P register and 0177777 in the A register. Pass 2 produces the binary object loader text and program listing and terminates when END is encountered with the

same register values as pass 1. A MORE directive causes the computer to stop and wait until the SI unit is prepared with the additional source input media, and RUN or START is pressed. MORE is indicated by 0170017 in the A register.

The program listing can be suppressed during pass 2 by resetting SENSE switch 2, and the binary output, resetting SENSE switch 3. Error messages cannot be suppressed and are output on the LO device as the error is detected during pass 2.

Synchronization errors (table 17-6) halt the assembly with 000777 in the A register. To continue the assembly, press RUN or START. The assembler resets the location counter value to that assigned on pass 1, prints error message *SE, and continues the assembly.

VARIAN 620 DAS ASSEMBLERS

Pass 2 can be restarted or repeated for extra copies of the assembled program without repeating pass 1.

At the completion of pass 2, the assembler can accept another assembly using the same I/O devices. For other I/O devices, reload the assembler program, starting with the I/O section.

To restart the assembler, press STEP, clear all registers, and press (SYSTEM) RESET and RUN or START. The assembler halts with 000001 in the P register and is ready to accept another assembly.

DAS 8A Operations

DAS 8A is designed to be used in computer systems with 8K or more of memory. It is supplied for use with user-specified I/O devices and has no I/O section for on-site device specifications. If a change of I/O devices is desired, a separate DAS 8A program is provided.

To operate under DAS 8A, load the assembler program into memory using BLD II (section 18). Prepare the I/O devices (SI, BO, and LO), and set the SENSE switches for pass 1 or 2 as described for DAS 4A assembler section operation.

Further operating procedures are the same as described above under Assembler Section Definitions for DAS 4A, except the I/O devices cannot be respecified.

DAS MR Operations

Since DAS MR operates under MOS and uses the MOS I/O control system, the I/O

devices can be defined as required (refer to the MOS manual, document number 98 A 9952 091).

DAS MR inputs the symbolic source statements from the processor input (PI) logical unit in alphanumeric mode, and outputs them in the same mode on the processor output (PO) logical unit. When DAS MR detects the END directive, it terminates pass 1, returns to the beginning of the source program, and begins pass 2. During pass 2, the source statements are the input from the system scratch (SS) logical unit, a listing is output on the LO unit, and the binary object program is output on the BO unit. Note that PO and SS must be the same magnetic tape, drum, or disc unit.

For an assembly without a program listing, input the following directive to the MOS executives when requesting the assembly:

/ASSEMBLE N

For a binary object program, input
/ASSEMBLE B

If the memory map portion (symbol table, external names, and entry names) is not wanted, input

/ASSEMBLE M

To read the same physical symbolic source statements for both assembly passes, input

/ASSIGN PO=DUM, SS=PI
/ASSEMBLE

The processor output listing serves as a copy of the program; it can be input for another assembly.

SECTION 18 - BINARY LOAD/DUMP PROGRAM

The **Varian 620 Binary Load/Dump Program (BLD II)** prepares the Varian 620-series computer for the loading of object programs from a high-speed or Teletype paper tape reader. It also allows the punching of the binary contents of memory on paper tape in a reloadable format.

BLD II is loaded using the bootstrap loader routine, which specifies the input reader. Once loaded, BLD II automatically relocates itself into the upper part of the highest 4K memory increment, unless the operator specifies another 4K increment. BLD II also dynamically adapts itself to load object program tapes from the input device specified in the bootstrap loader routine, and performs a check-sum of object program records.

After BLD II has been loaded into memory, it need not be reloaded for the entering of subsequent object programs.

Initially, BLD II occupies addresses 007000 through 007755 of the first 4K memory increment, where it does not interfere with the bootstrap loader routine occupying addresses 007756 through 007776. Immediately after loading, BLD II relocates to occupy addresses 0x7400 through 0x7755, where x denotes the highest or operator specified 4K of memory:

x =	Memory Increment
0	4K
1	8K
2	12K
3	16K
4	20K
5	24K
6	28K
7	32K

Entry to BLD II to load object program tapes is always 0x7600, and entry to punch binary object tapes of memory contents is 0x7404.

Loading the Bootstrap Routine

Table 18-1 lists the manual bootstrap loader routines. If the high-speed paper tape reader is to be used for subsequent program loading, select the column headed High-Speed Reader Code; for the Teletype paper tape reader, select the column headed Teletype Reader Code.

To load the bootstrap loader routine:

- Ensure that computer power is turned on and that the system is initialized.
- In step mode, load a store A register relative to P instruction (054000) into the instruction register.
- Set the REPEAT switch.
- Load the starting memory address of the bootstrap loader (007756) into the P register.
- Select the first bootstrap loader instruction from the appropriate column in table 18-1, and load it into the A register.
- Press STEP or START to load the A register contents into the address specified by the P register, which is incremented by one after the instruction is loaded.
- Clear the A register.

BINARY LOAD/DUMP PROGRAM

h. Repeat steps e, f, and g for each bootstrap loader instruction.

d. Load 007756 into the P register, keeping the REPEAT switch set.

e. Select the A register, and press STEP or START.

To verify bootstrap loading:

a. Initialize the system by pressing (SYSTEM) RESET.

b. Clear all registers.

c. Load 014000 (load A register relative to P) into the instruction register.

The contents of the memory addresses are displayed sequentially each time the STEP or START switch is pressed. If an error is found, load the correct instruction code into memory. **Note that the P register error address is always the error address plus one.**

BLD II, and subsequent object programs, can now be loaded into memory.

Table 18-1. Bootstrap Loader Routines

Address	High-Speed Reader Code	Teletype Reader Code	Symbolic Coding		
007756	102637	102601	READ	CIB	RDR
007757	004011	004011		ASLB	NBIT - 7
007760	004041	004041		LRLB	1
007761	004446	004446		LLRL	6
007762	001020	001020		JBZ	SEL
007763	007772	007772		(Memory address)	
007764	055000	055000		STA	0,1
007765	001010	001010		JAZ	LHLT + 1
007766	007000*	007000*		(Memory address)	
007767	005144	005144		IXR	
007770	005101	005101	ENTR	INCR	1
007771	100537	102601	SEL	SEL	RDON
007772	101537	101201		SEN	IBFR,READ
007773	007756	007756		(Memory address)	
007774	001000	001000		JMP	* - 2
007775	007772	007772		(Memory address)	

NOTE

The bootstrap loader routine is always loaded into the specified addresses of the first 4K memory increment, regardless of available memory.

* Replace this code with 007600 if the test executive of MAINTAIN II (refer to document number 98 A 9952 060) is to be loaded and executed.

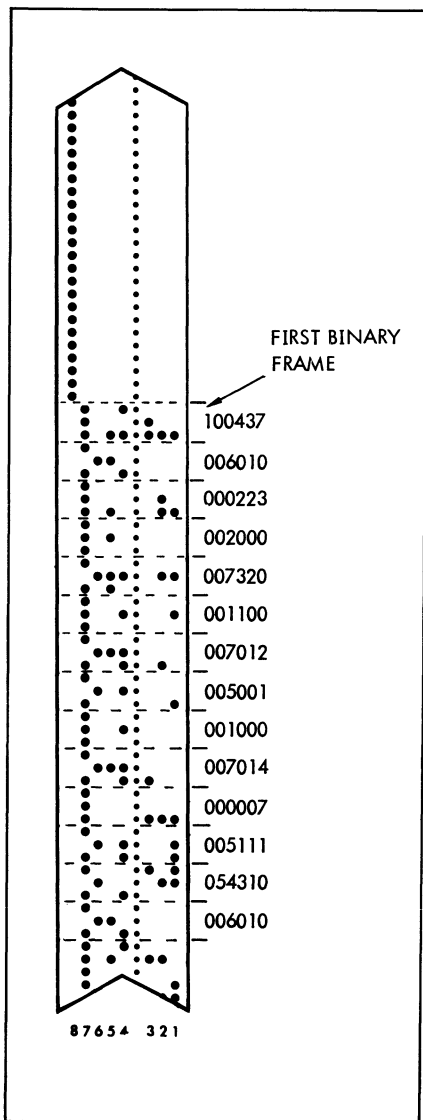
Loading the BLD II Program

CAUTION

To adapt to the input device, BLD II examines address 000200 to determine if the system includes the automatic bootstrap loader (ABL) option, then the contents of the first address of the manual bootstrap loader routine, both of which can indicate the input device. If address 000200 inadvertently contains one of the two input device codes, and the device used is different, BLD II malfunctions.

After the bootstrap loader routine has been successfully loaded into memory:

- a. Reset the REPEAT switch.
- b. Clear the instruction register.
- c. Load 007770 into the P register.
- d. Load 007000 into the X register.
- e. Set the SENSE switch(es) for the desired program option (table 18-2).
- f. Turn on the paper tape reader specified by the bootstrap loader routine.
- g. Position the BLD II program tape in the reader with the first data frame after the position-8-only punches (figure 18-1) under the high-speed reader head or under the reading station of the Teletype reader.
- h. In run mode, press RUN or START. Loading is complete when the computer changes to step mode.



VT11-1188.4

Figure 18-1. BLD II Tape Format
(Bootstrap-Loadable)

Table 18-2. BLD II SENSE Switch Options

SENSE Switch	When Set =
1	Allows selection of any 4K memory increment in which BLD II is to operate, or specification of a nonstandard device address for the high-speed paper tape punch.

After BLD II is loaded, the computer halts with 07013 in the P register.

To specify a 4K memory increment, load one of the following in the A register:

A Register	Memory Increment
000000	First 4K
000001	Second 4K
000002	Third 4K
000003	Fourth 4K
000004	Fifth 4K
000005	Sixth 4K
000006	Seventh 4K
000007	Eighth 4K

The standard high-speed paper tape punch device address is 037. To specify a nonstandard device address, load it into the B register.

Result: Pressing RUN or START initiates the relocation of BLD II from the first 4K memory increment and implements the punch address. The computer halts with zeros in the A, B, and X registers and 0x7600 in the P register, where x = the specified increment as described above. Object program tapes can then be loaded.

SENSE Switch	When Set =
2	Adjusts the program for Teletype paper tape punch output. (For use when input is from high speed reader, but a high speed punch is not available.)

Table 18-2. BLD II SENSE Switch Options (continued)

SENSE Switch

When Set =

3

Allows splicing an object program to the BLD II program tape.

Result: BLD II and the object program can be loaded and executed without further operator intervention.

NOTE

If no SENSE switches are set, the BLD II program is loaded and relocates automatically to the highest 4K memory increment. The computer then halts with the entry address for reading object program tapes in the P register (0x7600) and zeros in the A, B, and X registers.

If SENSE switch 1 was set:

- a. Reset SENSE switch 1.
- b. Clear the A register.
- c. Load the appropriate values, as defined in table 18-2, in the A and/or B registers.
- d. Press RUN or START.

When BLD II loading is complete, the computer halts with 0x7600 in the P register unless SENSE switch 3 was set (table 18-2), in which case the computer implements loading and execution of the spliced object program.

Remove the BLD II program tape from the reader after loading, and reset SENSE switch 2, if applicable.

Loading an Object Program

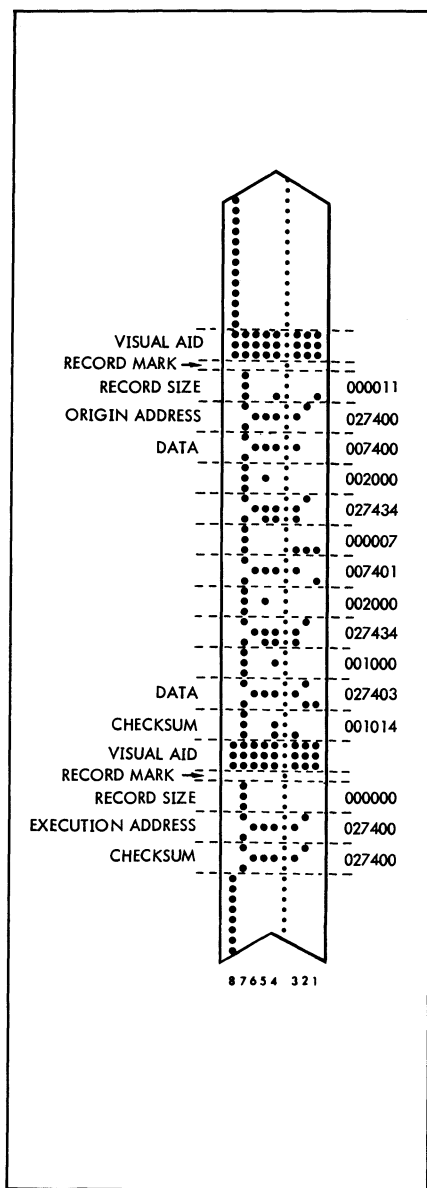
Object programs can be loaded from the bootstrap-routine-specified device immedi-

ately after BLD II. For all subsequent loadings, make sure that the P register is set to 0x7600.

Verification

To ensure that an object program tape contains no errors before it is loaded into memory, BLD II has a check-sum error-checking option. To use this option:

- a. Turn on the bootstrap-routine-specified reader.
- b. Position the object program tape in the reader with leader at the reading head (figure 18-2).
- c. Load minus value (0100000) into the A register.
- d. Clear the instruction register.
- e. In run mode, press RUN or START.



VT11-1189

Figure 18-2. Object Program Tape Format

No errors are indicated by the computer halting with:

P register = 0x7600
 A register = 0100000
 B register = 000000
 X register = execution address

If a check-sum error occurs, the computer halts with:

P register = 0x7600
 A register = 0100000
 B register = 0177777
 X register = Address of last record

To retry a check-sum error record, reposition the object program tape at the previous record mark and press RUN or START. If a check-sum error is again read, visually check each character in the record for an error in punching or damaged tape.

Load Program and Halt

To load the object program and halt before execution:

- Turn on the reader and position the tape in the reading station.
- Clear the A, B, X, and instruction registers.
- Load 0x7600 into the P register.
- In run mode, press RUN or START.

Correct loading is indicated when the computer halts with:

P register = 0x7600
 A register = 000000
 B register = 000000
 X register = execution address

A check-sum error is indicated by the conditions described for object program tape verification described above.

Load Program and Execute

Programs can be loaded and immediately executed using the steps described above for the load-and-halt option, except in step b load 000001 (or any positive number) in the A register.

Punching Program Tapes

The BLD II program adapts to the input reader and the output punch devices by interrogating the bootstrap loader routine. Setting SENSE switch 2 (table 18-2) prior to loading an object program adjusts the program for Teletype punch output regardless of the bootstrap-routine-specified devices.

To punch reloadable object program tapes after the programs have been loaded into memory, turn on the punch and:

- a. Load the beginning address of the area to be punched into the A register.
- b. Load the final address to be punched into the B register.
- c. Load the first instruction to be executed at load time into the X register,

OR

if noncontiguous memory areas are to be punched, load minus one (0177777) into the X register.

- d. Load 0x7404 (entry address to BLD II to punch object tapes) into the P register.

- e. Clear the instruction register.

- f. In run mode, press (SYSTEM) RESET and RUN or START.

The program punches the object tape and the computer halts with all registers unaltered.

If noncontiguous areas are to be punched, perform steps a through f. Prior to punching the last area, load the first instruction to be executed at load time into the X register.

Punching Memory Contents

To punch a tape of the binary memory contents on the high-speed paper tape punch, SENSE switch 2 must not be set when BLD II is loaded. To punch a tape from memory on the Teletype punch, SENSE switch 2 must be set (if the input reader is a high-speed paper tape device).

The operator can specify that tapes be punched in binary format for reloading using the BLD II, or that the BLD II program be punched in bootstrap-loadable format.

To punch a tape in binary format, use the procedures described above for punching program tapes.

To punch a bootstrap-loadable tape of BLD II itself:

- a. Load 0x7400 into the P register.
- b. Clear the A and B registers.
- c. Load a nonzero value into the X register.
- d. In run mode, press (SYSTEM) RESET and RUN or START.

SECTION 19 - AID II DEBUGGING PROGRAM

The **Varian 620 AID II Debugging Program** is supplied for use with all Varian 620-series systems. AID II provides software to facilitate on-line program checkout and correction. By entering AID II commands on the Teletype keyboard, the operator can:

- a. Display and alter the contents of registers and any memory address or group (block) of addresses.
- b. Transfer (trap) into or out of selected blocks of memory and search for specific conditions.
- c. Load, monitor, and alter any program.

As an added feature, data can also be transferred (dumped) from memory to magnetic tape, punched out on paper tape, or printed on the Teletype printer. Object programs can thus be converted from one media to another, simply and directly.

AID II is loaded into computer memory using the binary load/dump program (BLD II, section 18). Once loaded, AID II resides in memory addresses 0x6000 through 0x7377, where x denotes the highest available 4K memory increment, as follows:

x =	Memory Increment
0	4K
1	8K
2	12K
3	16K
4	20K
5	24K
6	28K
7	32K

The programmer is responsible for ensuring that a program to be debugged does not interfere with those areas of memory containing BLD II and AID II.

Loading AID II

To load AID II into memory:

- a. Ensure that the bootstrap loader routine and BLD II (section 18) are correctly loaded.
- b. Turn on the reader used to load BLD II and position the AID II program tape with leader at the reading station.
- c. Clear the B, X, and instruction registers, and load 000001 into the A register.
- d. Load 0x7600 into the P register (i.e., the BLD II entry address for loading program tapes; refer to section 18 for the definition of x).
- e. In run mode, press RUN or START.

Loading is complete when the program outputs a carriage return (CR) and line feed (LF) and rings the Teletype bell.

Programs to be debugged can be loaded either before or after AID II loading.

Register and Memory Modification

With AID II and the program to be debugged entered, the computer in run mode, and the Teletype operating on-line, the Teletype keyboard entries summarized in table 19-1 produce the indicated results.

The pseudoregisters referred to in the following descriptions denote software buffers that duplicate the actual contents of the computers's operation registers. A command to change register contents, in effect, changes the specified pseudoregister contents, which are then transferred to the corresponding operation register.

Table 19-1. AID II Register/Memory Modification Commands

Command	Operation
A	Displays (prints) the contents of the indicated pseudo-register on the Teletype printer. To change the contents, type the desired octal number and a period; otherwise, type only a period.
B	
X	
Cx	Displays (prints) the contents of memory address x on the Teletype printer. To change the contents, type the desired octal number, followed by a period to execute the command or by a comma to request display of the next sequential address contents. Otherwise, type only a period.
Gx.	Loads the contents of the pseudoregisters into the respective A, B, and X registers and starts program execution at address x.
Ix,y,z.	Stores the value of z in all memory addresses starting at address x and ending at address y.
Sx,y,z,m.	Searches through memory starting at address x and ending at address y for the value of z masked by the value of m. A masked-search compares the value of z with each bit corresponding to a one in the m value. Each time the values compare, the address and value are printed on the Teletype printer. If an N is typed instead of a mask value, the program searches for the negative value of z. Omission of m assumes an all-ones mask.
Ty,x.	Transfers execution of an operational program to address y when the program reaches the instruction in address x. This trapping feature permits interrupting a program sequence without internal patching. The program also displays the transfer address and the contents of the A, B, and X pseudoregisters, respectively.
Vx.	Displays the contents of memory on the Teletype printer beginning at address x, continuing until a RUBOUT character is typed. The display (dump) is printed in columns: the left column is the octal base address, and the contents of eight memory addresses, in ascending order, appear in the next eight columns. The first number in succeeding lines indicates the base address for the next eight memory address contents.

Usage Examples

NOTE

In the following examples, operator inputs are represented in bold type. Other entries are program responses output to the Teletype printer.

Display the contents of a pseudoregister:

```
A  142340 .
B  001000 .
X  006003 .
```

Display and change the contents of a pseudoregister:

```
A  010454  10406.
B  006016  10406.
X  007413  10406.
```

Display the contents of memory address 002050:

```
C02050    = 102401 .
```

Display and change the contents of memory address 002050, then display the next two addresses:

```
C02050    = 102401  103402,
( 002051 ) = 000067 ,
( 002052 ) = 177777 .
```

Display memory contents starting at address 006000:

```
V06000.
( 006000 ) 010454 002000 ...
( 006010 ) 005145 004543 ...
( 006020 ) 005041 001000 ...
( 006030 ) 006217 001000
```

NOTE

Eight columns of data actually follow the base address in the first column. Space limitations prohibit an actual representation herein.

(Display terminated by entering RUBOUT.)

Execute the program beginning at address 000500:

G500.

Store 0177777 in memory addresses 000200 through 000210:

I200,210,177777.

Search memory addresses 000200 through 000240 for a content of 0106213 masked by 0177777 and display addresses that compare:

```
S200,240,106213,177777.
( 000220 ) = 106213
( 000235 ) = 106213
```

Trap to memory address 000204; start execution from address 000100; and display the trap address and the A, B, and X register contents, if the trap is reached. **If not, reload the original contents into both trap locations.**

T204,100.

```
( 000204 ) 142340 002000 010405
```

Paper Tape Handling

The Teletype paper tape reader and punch can be controlled through AID II to read object program tapes into, and punch program tapes from, computer memory.

With AID II entered, the computer in run mode, and the Teletype and its paper tape system operational, the Teletype keyboard entries summarized in table 19-2 produce the indicated results.

Table 19-2. AID II Paper Tape Commands

Command	Operation
Dx,y,z.	Punches a program tape from the contents of address x through address y, specifying execution address z.
Lm.	Reads an object program paper tape into memory. If the value of m is 0 and no check-sum errors are encountered, the program is executed. If the value of m is 1 and no check-sum errors are encountered, the contents of the A, B, and X registers, respectively, are output on the Teletype printer: A register = 000000, B register = 000000, and X register = execution address. If the program detects a check-sum error, the computer halts with 0177777 (minus one) in the B register and the address of the last record read from the tape in the X register.
NOTE	
AID II utilizes BLD II to effect loading and punching. For proper operation, BLD II must reside in the same 4K increment of memory as AID II.	

Magnetic Tape Handling

Data can be manipulated from and to magnetic tape through AID II commands.

With AID II entered, the computer in run mode, the Teletype keyboard on-line, and the selected magnetic tape unit operational, the Teletype keyboard entries summarized in table 19-3 produce the indicated results.

In the following descriptions, x specifies the magnetic tape controller device address coded as 0, 1, 2, and 3, where 0 = first system magnetic tape controller, 1 = second system controller, etc. Note that each magnetic tape controller monitors up to four magnetic tape units and that AID II communicates only with the first unit on each controller.

Table 19-3. AID II Magnetic Tape Commands

Command	Operation
Ex.	Writes a file mark on the specified unit tape.
Fn,x.	Skips to file n on the specified unit tape.
N.	Skips to the next file on the previously designated unit tape.
Px.	Backspaces one record on the specified unit tape.
Rx	Reads an object magnetic tape into memory from the specified magnetic tape unit. Terminating the command with a period causes the program to be loaded and control returned to AID II. If the command is terminated with a comma, the program is loaded and executed. If AID II outputs an uparrow (↑) on the Teletype printer, a file mark was read on the tape. The output of an octal number indicates the address of a parity error.
Wa,b,c,x.	Writes an object magnetic tape from memory, starting at address a and ending at address b with an execution address of c, on the specified magnetic tape unit.

Error Message and Correction

If an AID II command is input incorrectly, AID II terminates further input by outputting a CR and LF and ringing the Teletype bell. An example of incorrect input is an attempt to type a nonoctal number (i.e., a decimal 8 or 9). Note that octal numbers

need not be preceded by a zero. To recover, correctly retype the entry.

An input command can be aborted before termination by the back slash (\) character.

Magnetic and paper tape error descriptions are included in tables 19-2 and 19-3.

SECTION 20 - SOURCE PROGRAM EDITOR

Varian's 620 Source Program Editor (EDIT) allows the Varian 620-series computer programmer to create and modify symbolic source programs (section 17) on paper tape. Source programs can be loaded directly into computer memory from an on-line Teletype keyboard, listed with identifying line numbers on the Teletype printer, and modified using EDIT commands input from the Teletype keyboard.

Source programs already formatted on paper tape can be loaded into memory, listed, modified with EDIT generating a paper tape of the modified program ready for assembly (section 17).

An added feature of EDIT is its ability to search through the source program and point to a specific character or group of characters, as well as entire lines and groups of lines.

EDIT has two modes of operation: command and text. In command mode, EDIT accepts inputs from the Teletype keyboard specifying the EDIT function and, optionally, line numbers and searching parameters. In text mode, characters typed on the Teletype keyboard or read from paper tape are stored in a text buffer for subsequent manipulation and/or output. The text buffer represents available memory, i.e., those memory addresses not occupied by the bootstrap loader routine, the binary load/dump program (BLD II, section 18), and the EDIT program routines.

EDIT operates in the minimum configuration of Varian 620 computer (4K to 32K of

memory) and 33/35 ASR Teletype. However, EDIT determines the size of memory and uses of all of available memory for the editing buffer; only the binary loader at the top of memory is saved. Use of the high-speed paper tape reader and/or punch for input/output is optional.

Loading EDIT

To load the EDIT program into memory:

- a. Ensure that the bootstrap loader routine and BLD II are correctly loaded (section 18).
- b. Turn on the reader used to load BLD II and position the EDIT program tape with leader at the reading station.
- c. Clear the B, X, and instruction registers.
- d. Load 000001 into the A register.
- e. Load 0x7600 into the P register (i.e., the BLD II entry address for loading object program tapes; refer to section 18 for the definition of x).
- f. In run mode, press RUN or START.

Loading is complete when the EDIT program outputs, on the Teletype printer, the message:

SOURCE PROGRAM EDITOR

SOURCE PAPER TAPE PROGRAM

INPUT DEVICE (H OR T)

If the high-speed paper tape system is to be used for text input to EDIT, type H on the Teletype keyboard, and type T if the Teletype is the input device. The program then outputs

OUTPUT DEVICE (H OR T)

Respond as described above for defining the input device. EDIT dynamically adapts to use the specified equipment and enters the command mode, outputting a carriage return (CR) and line feed (LF), followed by an asterisk (*), to the Teletype printer.

Once entered, EDIT can be restarted at any time by clearing all registers and pressing RUN or START.

NOTE

To change input and output devices from those initially specified, EDIT must be reloaded using the procedures described above.

EDIT Commands

With EDIT loaded, the computer in run mode, and the Teletype operating on-line, the Teletype keyboard entries summarized in table 20-1 produce the indicated results. Pressing the RETURN key terminates and executes all EDIT commands.

Table 20-1. EDIT Commands

Command	Operation
A	Enter text mode and add the following text input from the Teletype keyboard to the contents of the text buffer.
nC	Delete the line specified by n, and replace it with new text.
m,nC	Delete and replace lines m through n.
nD	Delete line n.
m,nD	Delete lines m through n.
F xxxx	Search the entire contents of the text buffer for character string xxxx (maximum number of characters, 72). Output sequential text lines until the string is detected and the line on which it appears is output. If the string is not found, return to command mode, and output CR, LF, and *.
nF xxxx	Go to line n and search it and succeeding lines for character string xxxx (see above).

Table 20-1. EDIT Commands (*continued*)

Command	Operation
G	List (output on the Teletype printer) the next sequential line whose first character is alphabetic.
nG	Go to line <i>n</i> and list the next line whose first character is alphabetic.
I	Insert the following text before the first line in the text buffer.
nI	Insert the following text before line <i>n</i> .
K	Delete the entire contents of the text buffer.
L	List the entire contents of the text buffer, assigning sequential line numbers (decimal), on the Teletype printer.
nL	List line <i>n</i> .
m,nL	List lines <i>m</i> through <i>n</i> .
P	Punch the contents of the text buffer on paper tape using the output device specified at EDIT loading time.
nP	Punch line <i>n</i> .
m,nP	Punch lines <i>m</i> through <i>n</i> .
R	Read (append) the following text input from the device specified EDIT loading time to the contents of the text buffer.
S	Search the contents of the text buffer for the character input after RETURN. Output sequential text lines on the Teletype printer until the line in which the character appears is printed. If the character is not found, return to command mode, and output CR, LF, and *.
nS	Go to line <i>n</i> and search for the character input after RETURN (see above).
m,nS	Search lines <i>m</i> through <i>n</i> for the character input after RETURN (see above).
T	Punch approximately 20 inches of leader/trailer on paper tape using the output device specified at EDIT loading time.

NOTES

Line numbers when specified in EDIT commands are decimal integers derived from the output of a listing command. The value of n must be greater than that of m.

Execution of all EDIT commands begins when the RETURN key is pressed.

Table 20-2 lists EDIT functions that are controlled by the use of Teletype special-

purpose keys. Note that their use differs in the two modes of operation.

Table 20-2. Teletype Key EDIT Functions

Teletype Key	Command Mode	Text Mode
RETURN	Execute the instruction	Load the input line into the text buffer
-	Illegal	Delete one character to the left
RUBOUT	Cancel the instruction	Delete all the line to the left and output \
CTRL and C (simultaneously)	Remain in instruction mode and output an asterisk (*)	Return to instruction mode and output an asterisk (*)
. (period)	Current line number (used alone or with the minus sign and number, e.g., 1. -8 refers to the eighth line preceding the current line)	Legal text character
/ (slash)	Number of the last line in the text buffer	Legal text character
=	Used with . and / to obtain their values	Legal text character
ESCAPE (ESC)*	List the next line	Ignored
CTRL and TAB (simultaneously)	Illegal	Interpreted as seven spaces on the Teletype printer output

* On the Model 35 Teletype, simultaneously press SHIFT, CTRL, and K.

Usage Example

To illustrate the use of EDIT commands and Teletype key functions, assume we wish to search line 20 for the character A and replace it with the character X. Note that the Teletype keys are shown enclosed in parentheses where they are applicable and that the simultaneous pressing of two or more keys is illustrated as follows: (SHIFT)(CTRL)(K).

- a. To ensure that EDIT is in command mode, type

(CTRL)(C)

- b. EDIT responds with a CR, LF, and *. Type

20S(RETURN)

- c. EDIT enters a delay loop and waits for input of the character for which it is to search. Type

A(RETURN)

- d. EDIT goes to line 20 and types it and succeeding lines until an A is found:

ORG
XYZ LDA

then waits for input. Type

-X(RETURN)

Other editing options available for use in step d are:

- a. To delete the line to the left, type RUBOUT.
- b. To delete the line to the right, type RETURN.

- c. To delete the entire line, type the appropriate deletion command (table 20-1).
- d. To delete characters from right to left, type - once for each character.
- e. To search for the next occurrence of the same character, simultaneously type CTRL and C.
- f. To continue searching, but for a different character, simultaneously type SHIFT, CTRL, and BELL, then input the new character.

Error Messages

EDIT checks all commands input to it for valid parameters and correct formatting. When an error is detected, EDIT:

- a. Types a question mark on the Teletype printer.
- b. Issues a CR and LF.
- c. Types an asterisk.
- d. Waits for a valid command.

The following conditions are recognized as errors:

- a. Incorrect response to the I/O device queries at loading time.
- b. A nonexistent command code.
- c. Commands terminated with any character other than RETURN.

SOURCE PROGRAM EDITOR

- d. A starting line number that is greater than an ending line number.
- e. Transposition of command parameters.
- f. Specifying a line number whose value is greater than the last line in the buffer.
- g. A deletion command that does not specify a line number.
- h. Pressing the ESCAPE (ESC) key to list the next line in the buffer when the buffer is empty.

When text being loaded into the text buffer exceeds the capacity of the buffer, EDIT outputs the message

BUF FULL

and returns to command mode. To save the buffer contents and continue processing:

- a. Type a punch (P) command (table 20-1) and RETURN.
- b. After punching is complete, restart EDIT by clearing all registers and pressing RUN or START.

The following options are also available:

- a. List, modify, and punch the buffer contents before restarting EDIT.
- b. Abort the current source program edit and continue processing with a new program.

SECTION 21 - MATHEMATICAL SUBROUTINES

In support of Varian 620 computer applications programs that require mathematical computations, Varian provides a comprehensive **Mathematical Subroutine Library** with complete, easily accessible subroutines.

The mathematical subroutines are grouped into four major categories: fixed-point arithmetic, floating-point arithmetic, arithmetic functions (both real and complex), and number and character conversions. The subroutines are called by other programs and fill the mathematical requirements of virtually all computer applications.

The mathematical subroutine library is described in detail in the Varian 620 Subroutine Descriptions Manual (document number 98 A 9902 044).

Fixed-Point Arithmetic

The fixed-point arithmetic subroutines are for applications that demand a high-speed arithmetic package. They include:

- a. Addition, subtraction, multiplication, and division (single- and double-precision)
- b. Two's complement (double-precision)
- c. Absolute value
- d. Transfer of sign

Fixed-point, single-precision multiplication (XMUL) provides a software version of the hardware option for multiplication. XMUL

uses successive addition of the multiplicand with appropriate left-shifts.

Fixed-point, single-precision division (XDIV) provides a software version of the hardware option for division. XDIV uses an unsigned, nonrestoring division algorithm.

Fixed-point, double-precision addition (XDAD) adds the double-precision number whose address is in the calling sequence to the double-precision number in the A and B registers. The low-order halves of the numbers are added first, and, if there is a carry, it is added to the high-order sum.

Fixed-point, double-precision subtraction (XDSU) subtracts the double-precision number whose address is in the calling sequence from the double-precision number in the A and B registers.

Fixed-point, double-precision multiplication (XDMU) multiplies the double-precision number whose address is in the calling sequence by the double-precision number in the A and B registers. XDMU uses double-precision addition of partial products.

Fixed-point, double-precision division (XDDI) divides the double-precision number in the A and B registers by the double-precision number whose address is in the calling sequence. XDDI returns the difference to the A and B registers.

Fixed-point, double-precision two's complement (XDCO) takes the two's complement of the double-precision number in the A and B registers. XDCO complements the number, then tests the low-order bits for a carry.

MATHEMATICAL SUBROUTINES

Fixed-point, integer absolute value (IABS) takes the absolute value of the signed integer in the A register. If the number is negative, IABS one's complements it, then corrects it to two's complement form.

Fixed-point, integer sign transfer (ISIG) applies the sign of the integer whose address is in the calling sequence to the quantity in the A and B registers.

Floating-Point Arithmetic

The floating-point subroutines provide higher accuracy, more flexibility, and wider number ranges than fixed-point arithmetic. Floating-point subroutines include:

- a. Addition, subtraction, multiplication, and division
- b. Absolute value
- c. Sign copy
- d. Mantissa separation
- e. Normalization

Floating-point addition (\$QK) algebraically adds the floating-point number in the A and B registers to the floating-point number whose address is in the calling sequence.

Floating-point subtraction (\$QL) computes the difference of the floating-point minuend in the A and B registers and the floating-point subtrahend whose address is in the calling sequence.

Floating-point multiplication (\$QM) multiplies the floating-point number in the A and B registers by the number whose address is in the calling sequence. \$QM separates the mantissa and calls XDMU to implement the arithmetic operation.

Floating-point division (\$QN) divides the floating-point number in the A and B registers by the number whose address is in the calling sequence. \$QN separates the mantissa and calls XDDI to implement the arithmetic operation.

Floating-point, real-number absolute value (ABS) takes the absolute value of the floating-point, real quantity in the A and B registers. If the number is negative, ABS one's complements it and returns the result in the A and B registers.

Sign copy (SIGN) sets the sign of the floating-point number in the A and B registers equal to the sign of the quantity whose address is in the calling sequence.

The mantissa separation subroutines (\$FMS, \$FSM) separate the floating-point number in the A and B registers and return the mantissa in the A and B registers and the characteristic in the X register.

Normalization (\$NML) normalizes the floating-point, double-precision number in the A and B registers. \$NML tests the sign, two's complements the number using XDCO, and returns the fixed-point result in the A and B registers and the sign flag in the X register.

Arithmetic Functions

Subroutines are provided for the following arithmetic functions:

- a. Logarithm
- b. Exponential function
- c. Square root
- d. Sine

e. Cosine

f. Arctangent

g. Polynomial

h. Exponentiation

Fixed-point, single-precision logarithm (XLOG) computes the natural logarithm of the quantity in the A register. XLOG uses a Chebyshev polynomial of the fifth degree.

Floating-point, double-precision logarithm (ALOG) computes the natural logarithm of the quantity whose address is in the calling sequence, returning the result in the A and B registers.

Fixed-point, single-precision exponential function, positive argument (XEXP) computes the exponential of the positive quantity in the A register.

Fixed-point, single-precision exponential function, negative argument (XEXN) computes the exponential of the negative quantity in the A register.

Floating-point exponential function (EXP) computes the exponential of the floating-point quantity whose address is in the calling sequence.

Fixed-point, single-precision square root (XSQT) takes the unrounded square root of the quantity in the A register (if it is nonnegative) and returns the result in the A register.

Floating-point square root (SQRT) takes the square root of the floating-point number whose address is in the calling sequence.

Fixed-point, single-precision sine (XSIN) computes the sine of the quantity in the A register, returning the result in the A register.

Floating-point sine (SIN) computes the sine of the floating-point quantity whose address is in the calling sequence.

Fixed-point, single-precision cosine (XCOS) takes the cosine of the quantity in the A register and returns the result in the A register.

Floating-point cosine (COS) takes the cosine of the floating-point quantity whose address is in the calling sequence.

Fixed-point, single-precision arctangent (XATN) computes the arctangent of the quantity in the A register, returning the result in the A register.

Floating-point arctangent (ATAN) computes the arctangent of the floating-point quantity whose address is in the calling sequence.

Fixed-point, single-precision polynomial (POLY) supports the fixed-point, single-precision mathematical subroutines that require the evaluation of a polynomial in one variable of any finite degree. The polynomial is evaluated in Horner form

Fixed-point, integer exponentiation (\$HE).

Integer/floating-point exponentiation (\$PE).

Floating-point exponentiation (\$QE).

Conversions

The number and character conversion subroutines include:

- a. Fixed-point/floating-point
- b. Binary/decimal
- c. EBCDIC/Hollerith
- d. EBCDIC/ASCII

Fixed-point, single-precision integer to floating-point conversion (\$QS) converts the signed integer in the A register to floating-point format.

Floating-point to fixed-point, single-precision integer conversion (\$HS) converts the floating-point number in the A and B registers to integer format.

Fixed-point, single-precision binary-to-decimal conversion (XBTD) converts the absolute value of the integer in the A register to a four-digit decimal-coded integer in the B register.

Fixed-point, single-precision decimal-to-binary conversion (XDTB) converts the four-digit, binary-coded-decimal integer in the A register to a pure binary integer in the B register.

EBCDIC-to-Hollerith conversion (SA01) converts and eight-bit EBCDIC character in the A register to its equivalent 12-bit Hollerith code, returning the result in the A register.

Hollerith-to-EBCDIC conversion (SB01) converts a 12-bit Hollerith code in the A register to its equivalent eight-bit EBCDIC character, returning the result in the A register.

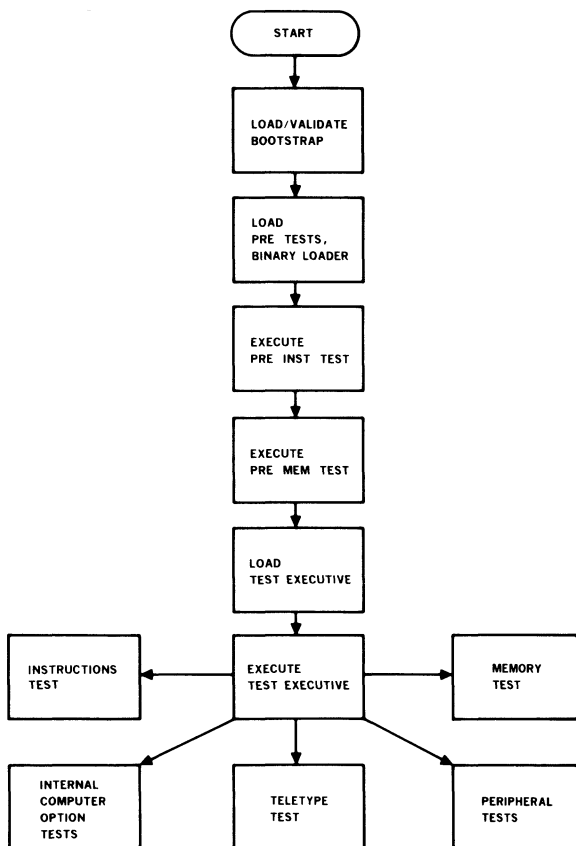
EBCDIC-to-ASCII conversion (SC01) converts an eight-bit EBCDIC character in the A register to its equivalent eight-bit ASCII code, returning the result in the A register. This subroutine can be modified to produce seven-bit ASCII codes.

SECTION 22 - MAINTAIN II TEST PROGRAMS

The **Varian 620 MAINTAIN II Test Programs** comprise a system approach to testing and maintaining Varian 620-series computers. MAINTAIN II is designed to verify total system operation, and to minimize maintenance time by aiding in

the isolation of system malfunctions to a specific area.

The major functional elements of the MAINTAIN II system (illustrated in figure 22-1) are:



NOTE: The Test Executive operates with only one test program in memory at a time.

VT11-0946A

Figure 22-1. MAINTAIN II Test Program System

MAINTAIN II TEST PROGRAMS

- Test executive
- Preliminary and comprehensive CPU and memory test programs
- Computer and I/O option test programs
- Peripheral and I/O interface test programs

MAINTAIN II is described in detail in the Varian 620 Test Programs Manual (document number 98 A 9952 060).

Test Executive Program

The test executive program controls the MAINTAIN II system. It includes preliminary CPU (instructions) and memory tests, a binary loader, and the test executive.

The preliminary instructions test portion of the test executive program validates basic CPU operation, the preliminary memory test monitors the operation of the first 4K memory increment, and the binary loader reads object program data and stores it in memory.

The test executive:

- a. Provides directives to control testing activities
- b. Loads and executes the other MAINTAIN II test programs
- c. Contains a utility subroutine package of aids for debugging, maintenance, and troubleshooting
- d. Includes standard test subroutines (Teletype input/output, program delays, SENSE switch options, etc.)

The operator communicates with the test executive through the Teletype keyboard and printer. Directives and parameters input from the keyboard control execution and monitoring of associated test programs. To accommodate the minimum 4K memory system, the test executive operates with only one test program in memory at a time.

The utility subroutines of the test executive provide the software to:

- a. Display and alter memory and register contents
- b. Search memory for specific data patterns
- c. Set areas of memory to various data patterns
- d. Punch paper tapes in both binary and object formats
- e. Interrupt test programs during their execution

Test Programs

MAINTAIN II test programs exercise the computer, options, and associated peripherals with sequences of instructions. If an instruction produces incorrect results, the sequence is halted and error messages indicate the failing instruction or operation. The operator can then repeat, continue, or halt the program until the fault is isolated. Good maintenance procedures include:

- a. Either on a routine preventive basis or when a system malfunction is suspected, run the MAINTAIN II test programs and eliminate from consideration the functional areas that are

MAINTAIN II TEST PROGRAMS

operating properly to isolate a fault to a specific area.

- b. Exercise the area of a suspected fault by executing, repeating, or modifying the applicable test program.
- c. Correct the fault by replacing the faulty component or circuit card, and restore the system to normal operation.
- d. Verify system operation by rerunning the test program.

The maintenance and reference manuals appropriate to the system installation describe theory of operation, timing, and signal locations and levels for the system components. Also given are system check-out procedures using the computer control panel and recommended test equipment.

MAINTAIN II test programs are normally supplied on punched paper tape; other media, such as card decks, are available. Loading and operating procedures are delineated in the Varian 620 Test Programs Manual (document number 98 A 9952 060).

SECTION 23 - MASTER OPERATING SYSTEM

The **Varian 620 Master Operating System (MOS)** is a batch-processing operating system for Varian 620-series computers. It is a complete integrated software package that operates in a wide range of hardware configurations.

MOS is modular, thus facilitating expansion and making optimum use of memory since only those portions of the system required for a specific operation need be loaded.

Features of MOS include:

- Extensive job control language -- 22 directives
- Flexible language processing
- File maintenance and editing programs and debugging aids
- Multisource storage, input, and output
- Comprehensive status and error reporting

MOS requires, in addition to the computer, a minimum of 8K of memory, a 33/35 ASR Teletype, and either a rotating memory (drum or disc) unit on a buffer interlace controller (BIC) or a magnetic tape unit. MOS supports and is enhanced by the addition of the high-speed paper tape and/or card reader, line printer, the hardware multiply/divide and extended addressing feature, high-speed paper tape and/or card punch, and additional memory increments.

MOS is described in detail in the Varian 620 Master Operating System Manual (document number 98 A 9952 091).

MOS is divided into resident (in memory) and nonresident partitions. Figure 23-1 shows the relationship.

The resident partition comprises the resident monitor, absolute loader, I/O assignment tables, system flags and parameters, and dump routine.

The nonresident partition comprises the control programs, support programs, and language processors, all of which need be in memory only when required for processing.

The **control programs** are:

- a. Executive
- b. System loader
- c. I/O control

The *executive program* directs execution of the user's programs. It interprets the system directives and either executes the instructions directly, or calls and initiates the appropriate software. Upon completion of the sequence, the executive resumes control and goes to the next directive.

The *system loader* can load program components unconditionally and/or selectively by program name. It accepts only relocatable object text and permits literal addressing and linking of external pro-

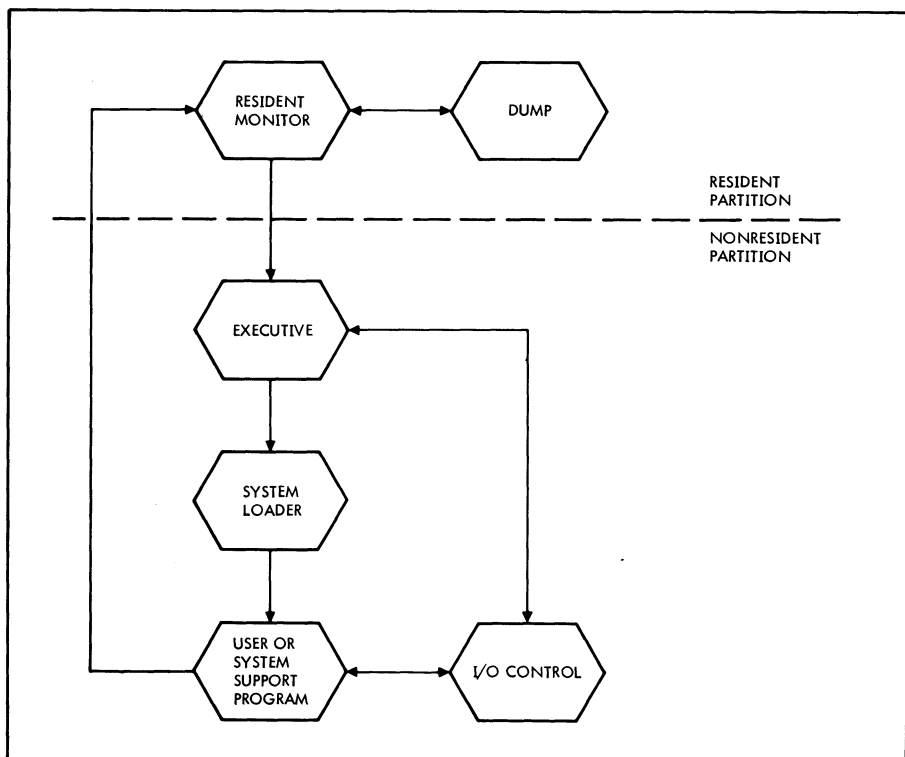
MASTER OPERATING SYSTEM

grams. When the system loader successfully completes an operation, it returns control to the resident monitor for program execution.

The *I/O control program* processes all I/O requests. I/O control is flexible and device-independent. This program assigns I/O functions to peripherals by logical unit designations instead of addressing the devices as hardware units. Up to 225 logical units can be so assigned, and they can be reassigned at any time. Thus, different peripherals can be substituted for I/O operations without reassembling the program.

MOS support programs include:

- a. The *debugging program* -- aids the programmer in finding and correcting program errors. Debugging commands permit examining and/or changing a program, in addition to running part or all of it.
- b. The *concordance program* -- analyzes the symbols of a DAS MR assembler program, indicating where they are defined and referenced. Any source program in the DAS language can be so analyzed.



VT11-0926

Figure 23-1. MOS Partitions and Flow

- c. The *file editing program*, or source editor -- provides for the creation, duplication, and correction of source files, such as symbolic DAS and FORTRAN IV program statements.
- d. The *file maintenance program* -- edits the MOS mathematical and support subroutine library. Program subroutines can be added to or deleted from the library.
- e. The *system preparation program* -- a stand-alone support program. It creates a system file on a magnetic tape

or rotating memory unit, tailored to the hardware and software requirements of the MOS installation.

- f. The *mathematical and support library*.

The MOS language processors are: the **DAS MR assembler**, and the **FORTRAN IV** compiler (requires an additional memory increment). Under MOS supervision, the processors automatically process source statements to generate relocatable and compatible object programs.

SECTION 24 - FORTRAN IV

The **Varian 620 FORTRAN IV Programming System** permits simple solutions of complex mathematical problems and is especially useful for scientific and engineering applications. The name FORTRAN is derived from the primary use of the language: **formula translation**.

Varian's FORTRAN IV system comprises a programming language, a library of subprograms, a compiler, and runtime program. It is available in both stand-alone and master operating system (MOS) configurations. The FORTRAN IV language is compatible with, and encompasses the capabilities of, American National Standards Institute (ANSI) FORTRAN, including its mathematical subroutine provisions.

A major feature of FORTRAN IV is its simplicity. Problems can be stated in simple English words and mathematical terms. Thus, persons having little computer organization experience and minimum programming skills can write effective programs after only a few hours of training.

The FORTRAN IV language consists of a series of source statements divided into physical sections called lines and coded to a precise grammatical format. The compiler analyzes the source program statements and transforms them into an object program suitable for execution on the Varian 620-series computers. One-pass processing provides convenient, efficient compilations, and the stand-alone system operates in only 8K of memory. The MOS version requires 12K.

The FORTRAN IV system is described in detail in the Varian 620 FORTRAN IV Reference Manual (document number 98 A 9902 037).

Programming With FORTRAN IV

The writing of effective FORTRAN IV programs is facilitated by the following features of the language:

- a. A *scale factor* allows internal and external representations of data to be modified during conversion.
- b. *Variable and array attributes* can be explicitly named by statements specifying:
 - (1) The number of words assigned to an item.
 - (2) A variable as integer, real, double-precision, complex, or logical.
 - (3) Array dimensions.
 - (4) Data initialization values for variables.
- c. *Subprogram array dimensions* can be specified as variables, and absolutes substituted when the subprogram is called.
- d. An *array can have one, two, or three dimensions*.
- e. A *variable name can contain up to six characters*.
- f. *Mathematical function subprograms* return results via an argument list.

FORTRAN IV

There are two classes of FORTRAN source statements: executable and nonexecutable. Executable statements specify program action. Nonexecutable statements describe program usage, operand characteristics, editing information, statement functions, and data arrangement. Functional groupings of these statements include data specification, arithmetic/logical expressions and assignments, input/output, subprogram definition, and program control.

Operational Features

The one-pass FORTRAN IV compiler translates source programs to relocatable machine-language object programs. Error diagnostics, source listings, and object listings are generated. Input/output and listing options can be selected for each program to be processed.

The FORTRAN IV relocatable loader enters into memory the object programs produced by the compiler and FORTRAN-compatible subprograms produced by the DAS 8A assembler (section 17). Object program media can be either paper or magnetic tape. Memory maps and error diagnostics are output on the Teletype printer.

Standard FORTRAN IV library subprograms include:

- a. Runtime input/output and utility
- b. Single- and double-precision and complex mathematical functions
- c. Single- and double-precision mathematical library

SECTION 25 - BASIC LANGUAGE

The **Varian 620 BASIC Language** is a popular, easy-to-use programming system for a wide variety of business and scientific applications. BASIC has many of the characteristics of ordinary mathematical notation: simple vocabulary and grammar, plus problem-solving steps can be specified completely and precisely.

The simplicity of BASIC, and its conversational operation, permit the inexperienced programmer to write and execute useful computer programs with a minimum of training. The computer responds to all commands entered by the operator, thus reinforcing the learning process. If the operator makes an error, diagnostics report the type of error, and corrections can be made immediately.

BASIC (Beginner's All-Purpose Symbolic Instruction Code) was originally developed at Dartmouth College. Varian's version of BASIC adapts it for use on the Varian 620-series computer systems.

For the experienced programmer, the Varian version of BASIC includes expanded instructions and capabilities. The advanced features permit wider-range programming applications, while retaining the inherent simplicity of the language.

Only 8K of memory and a Teletype are required for using BASIC in Varian 620-series computer systems. Even dedicated computers can perform general computation when they are not being used for other primary functions.

BASIC is described in detail in the **Varian 620 BASIC Language Manual** (document number 98 A 9952 032).

Programming With BASIC

The simplicity of programming with BASIC is illustrated by the following features of the language:

- a. Each line of the BASIC program begins with a **line number** that identifies the statement, and specifies the order in which the statement is to be processed by the computer. The program can be written in any order since the computer sorts and edits it as specified by the line numbers.
- b. Each statement has a word following the line number. This word specifies the **type of statement** and, thus, the operation to be performed by the computer in self-defining terms.
- c. BASIC uses only **capital letters** corresponding to the letters of the Teletype keyboard, which is used to input the programs.
- d. Each statement is **free-form**. Thus, statement fields and spacing can be disregarded during input.
- e. BASIC uses the following five **arithmetic operators**, each indicated by the corresponding symbol:

Addition	+
Subtraction	-
Multiplication	*
Division	/
Exponentiation	↑

BASIC LANGUAGE

the following **arithmetic relationships**

Equal to	=
Not equal to	#
Less than	<
Greater than	>
Not less than	> =
Not greater than	< =

the following **logical operators**:

AND OR NOT

a wide variety of **mathematical and special functions**, and complete **matrix operations**.

Operational Features

BASIC can be operated in two modes: program and calculator. Program mode is defined as entering a set of numbered BASIC statements that are checked for validity and stored, then entering the control command RUN to

start execution of the program. The calculator mode causes the computer to respond immediately to a BASIC statement; in this mode, the operator enters a BASIC statement, but not a statement number, and the BASIC system executes it immediately.

BASIC also features:

- a. Immediate syntax-checking, statement diagnosis, and error correction.
- b. Optional Teletype or high-speed paper tape input/output.
- c. Optimum use of memory through operator library selection.
- d. Control commands to halt and begin program execution, for input/output selection, and for deleting a program from memory.

SECTION 26 - RPG IV (REPORT PROGRAM GENERATOR)

The **Varian 620 RPG IV (Report Program Generator)** language is an advanced version of the widely used RPG commercial and general data-processing systems. RPG IV permits the concise coding of powerful programs, simply and efficiently. Thus, users with background other than data-processing can use RPG IV problem-solving techniques without extensive training or practice.

RPG IV programs are far more concise than equivalent programs written in the COBOL language. In typical instances, only one-fourth to one-third of the program steps are required. This results in more efficient processing and reduces the amount of memory required to store the program.

Varian's RPG IV improves on basic RPG in that it incorporates many automatic features and powerful procedural statements. Each RPG IV statement is written free-form, thus simplifying the programmer's task.

RPG IV is particularly adapted to processing data for the output of reports, but has many other applications as well.

The RPG IV system is offered in a stand-alone version, which operates with a Varian 620-series computer with a 4K (minimum) memory, card reader and punch, and line printer. RPG IV can also be operated under the control of the Varian 620 Master Operating System (MOS, document number 98 A 9952 091).

RPG IV is described in detail in the Varian 620 RPG IV System User's Manual (document number 98 A 9947 031).

Programming With RPG IV

RPG IV programs comprise two sequences of statements. First, there is a sequence of data-defining statements delineating the structures and formats of the data to be processed. This is followed by a sequence of procedural statements; these statements process the data through the structures defined in the first sequence. These two sequences of statements handle tables and records, update files, produce reports, and can deal with any other business-oriented applications.

Four types of **data-defining statements** provide a definition of the data structures to be used by the program. These data structures are records and tables. Records hold intermediate results and data being input from or output to files. Tables contain related, repetitive data items.

Both records and tables are divided into fields. Fields are the elementary variables of any RPG IV program. The computations performed by the program, its logic, and its final output are based on the manipulation of fields and their contents.

The functions of data-defining statements are:

- a. A **record statement** -- identifies a record and specifies the conditions under which this record is manipulated.
- b. A **record field statement** -- identifies and defines all of the fields in the record. All record field statements pertaining to a given record immediately follow the record statement for that record.

- c. A **table statement** -- identifies a table and specifies its size.
- d. A **table field statement** -- identifies and defines all of the fields in the entries in a table. All table field statements pertaining to a given table immediately follow the table statement for that table. Each entry in a given table has the same field structure as any other entry in that table.

Procedural statements follow the data-defining statements. They direct the execution of the program as it processes the data previously defined by the data-defining statements.

Procedural statements are executed in the order of their appearance in the program unless a specified condition is not met, or unless the program is directed to another statement by the statement in process.

Each RPG IV program statement can be numbered so that the program has access to it as required. Numbering is optional for statements that do not require other than sequential access.

Another important feature of RPG IV is that comment lines can be included to

clarify a program, improve the format of the output listing, or document the program.

Operational Features

The basic component of the Varian 620 RPG IV System is the two-part compiler. This compiler accepts RPG IV source program statements and, in one pass, produces both a listing of the program and an object deck. The listing includes diagnostics and error messages. The object deck, the RPG IV loader, and the RPG runtime support program process the data given in the data-defining portion of the source program.

The principal output of RPG IV is a printed record output on the line printer, and ready for reproduction and/or distribution. In certain applications, the program outputs a portion or all of the processed data on punched cards to be used as input for later data processing. For example, the record of an updated end-of-the-month inventory can be used as start-of-the-month data at the end of the following month.

Specific operating procedures for both the stand-alone and MOS-controlled versions of RPG IV are given in the Varian RPG IV Manual (document number 98 A 9947 031).

SECTION 27 - VORTEX

The **Varian Omnitask Real-Time EXecutive (VORTEX)** is a modular software operating system for controlling, scheduling, and monitoring tasks in a real-time multiprogramming environment. VORTEX also provides for background operations such as compilation, assembly, debugging, or execution of tasks not associated with the real-time functions of the system.

The basic features of VORTEX comprise:

- Real-time input/output processing
- Provision for directly connected interrupts
- Interrupt processing
- Multiprogramming of real-time and background tasks
- Priority task scheduling (clock time or interrupt)
- Load and go
- Centralized and device-independent input/output
- Operator communications
- Batch-processing job-control language
- Program overlays
- Background programming aids: FORTRAN compiler, DAS MR assembler, load-module generator, library updating, debugging, and source editor

- Use of background area when required by foreground tasks
- Disc/drum directories and references
- System generator

A rotating-memory device (either disc or drum) serves as storage for the VORTEX operating system components, enabling real-time operations and a multiprogramming environment for solving real-time and nonreal-time problems. Real-time processing is implemented by hardware interrupt controls and software task scheduling. Tasks are scheduled for execution by operator requests, other tasks, device interrupts, or the completion of time intervals.

Background processing (nonreal-time) operations, such as FORTRAN compilations or DAS MR assemblies, are under the control of the job-control processor, itself a VORTEX background task. These background processing operations are performed simultaneously with the real-time foreground tasks until execution of the former is suspended, either by an interrupt or a scheduled task.

All VORTEX tasks are scheduled, activated, and executed by the real-time executive component on a priority basis. Thus, in the VORTEX operating system, each task has a level of priority that determines what will be executed first when two or more tasks come up for execution simultaneously.

The job-control processor component of the VORTEX system manages requests for

VORTEX

the scheduling of background tasks.

Upon completion of a task, control returns to the real-time executive. In the case of a background task, the real-time executive schedules the job-control processor to determine if there are any further background tasks for execution.

During its execution, any foreground task can use any real-time executive service, i.e., operations that the task itself cannot perform including those involving linkages with other tasks.

VORTEX requires the following minimum hardware configuration:

- a. Varian 620/f-100 computer with 12K read/write main memory (16K foreground and background)
- b. Direct memory access (DMA)
- c. 33/35 ASR Teletype on a priority interrupt module (PIM)
- d. Real-time clock (RTC)
- e. Memory protection (MP)
- f. Power failure/restart (PF/R)
- g. Optional instruction set (OIS)

h. PIM

- i. Rotating-memory device on a PIM with either a buffer interlace controller (BIC) or priority memory access (PMA)
- j. One of the following on a PIM:
 - (1) Card reader with a BIC
 - (2) Paper tape system or a paper tape reader
 - (3) Magnetic tape unit with a BIC

The system supports and is enhanced by the following optional hardware items:

- a. Additional main memory (up to 32K) and/or rotating memory
- b. Automatic bootstrap loader (ABL)
- c. Card reader, if one is not included in the minimum system
- d. Card punch with BIC and PIM
- e. Line printer with BIC and PIM
- f. Paper tape punch, if one is not included in the minimum system.

The VORTEX system is described in detail in the VORTEX Reference Manual (document number 98 A 9952 100).

SECTION 28 - VARIAN 620/L-100 COMPUTER

The **Varian 620/L-100 Computer** is a general-purpose digital computer, designed for a variety of system applications.

The computer processes 16-bit words in a full-cycle execution time of 950 nanoseconds, or over one million cycles per second.

The instruction set of the Varian 620/L-100 comprises 115 standard, and 18 optional, instructions, many of which can be microcoded to extend the effective repertoire to several hundred instructions. Optional instructions include hardware multiplication and division, and indirect, immediate, and extended addressing operations.

Core memory can be expanded in 4,096-word (4K) increments, from a minimum of 4K to a maximum of 32K. Improved design allows the packaging of a fully expanded 32K system in two 10-1/2-inch high, standard rack enclosures.

The central processing unit (CPU) features four user-accessible operation registers, five buffer registers, an overflow indicator, and convenient operator's control panel.

Six addressing modes can be implemented: direct, multilevel indirect, immediate, indexed, relative, and (optionally) extended forms that permit direct addressing of any area of the fully expanded 32K system.

One power supply can furnish all the power required to maintain the maximum 32K

system plus a number of peripheral controllers.

The computer mainframe chassis accommodates the circuitry for the CPU, an 8K master memory, all the available mainframe (internal) option, and up to nine peripheral controllers.

Mainframe standard features include: hardware multiply/divide and extended addressing (M/D), memory protection (MP), real-time clock (RTC), and power failure/restart (PF/R).

The standard Varian 620/L-100 party-line input/output (I/O) bus can interface a maximum of ten peripheral controllers. Additional peripheral controllers can be accommodated by including an I/O buffer card.

System I/O options include: priority interrupt module (PIM) and buffer interlace controller (BIC). The PIM establishes eight levels of interrupt priority for selected peripheral controllers and places interrupt requests on the I/O bus in order of priority. The BIC implements the direct memory access (DMA) capabilities of the basic computer, permitting cycle-stealing I/O data transfers between memory and peripheral controllers at rates of up to 382,720 words per second.

Varian offers a complete complement of peripheral devices and their controllers to simplify total system planning and installation. Available peripherals include: high-speed paper-tape equipment, magnetic-tape units, disc and drum memories, punched-card equipment, and a variety of

VARIAN 620/L-100 COMPUTER

analog, digital, and communications equipment.

Standard software for the Varian 620/L-100 includes: the DAS symbolic assemblers, FORTRAN IV, BASIC, Master Operating System (MOS), the business-oriented RPG IV, AID II for debugging, MAINTAIN II for hardware troubleshooting, plus a complete library of mathematical and utility subroutines.

Varian's field-service organization provides a comprehensive program to aid the user of the Varian 620/L-100 in planning, installing, and maintaining the total system application. Service contracts cover necessary scheduled preventive maintenance and emergency repairs. VOICE, Varian's user organization, acts as a clearinghouse for user inquiries and suggestions, and provides information about new products and system applications, both hardware and software.

SECTION 29 - VARIAN 620/L-100 SPECIFICATIONS

Description	System-oriented, general-purpose digital computer for on-line data processing	
Memory	Magnetic core, with a 16-bit word length, 950-nanosecond full-cycle time, 425-nanosecond access time, and expandable from the basic 4,096-word (4K) minimum to a maximum of 32,768 words (32K)	
Arithmetic	Parallel, binary, fixed-point, two's complement	
Word Length	16 bits	
Machine Cycle Speed (Fetch and Execute)	Addition/subtraction:	1.9 microseconds
	Multiplication (optional):	9.5 microseconds
	Division (optional):	9.5-13.2 microseconds
	Register modification:	0.95 microseconds
	A/B register input/output:	1.9 microseconds
	Memory input/output:	2.85 microseconds
Instruction Set	115 standard, and 18 optional, instructions, many of which can be microcoded for extended operations	
Instruction Types	One- and two-word addressing, and one- and two-word nonaddressing instructions performing the following functions:	
	Load/store	Jump
	Shift/rotation	Jump and mark
	Register modification	Execution
	Arithmetic	Control
	Logic	Input/output
Addressing Modes	Direct, to 2,048 words	
	Relative to P register, to 512 words	
	Indexed with X or B register, to 32,768 words (does not add to execution time)	
	Multilevel indirect	
	Immediate	
	Extended (optional)	

VARIAN 620/L-100 SPECIFICATIONS

Operation Registers	A register: 16-bit accumulator and shift register
	B register: 16-bit accumulator and shift register (low-order half of the double-length accumulator), or index register
	X register: 16-bit index register
	P register: 16-bit program counter
Auxiliary Registers	U register: 16-bit instruction register
	L register: 15-bit memory address register
	W register: 16-bit memory data register
	S register: 5-bit shift register
	R register: 16-bit operand register
Control Panel	Register entry switches and display indicators; overflow (OVFL), STEP, and RUN indicators; REGISTER select and RESET switches; three SENSE switches; instruction REPEAT, STEP, and RUN switches; SYSTEM RESET, and three-position power switch
Logic and Signals	Integrated circuits and 4.211 MHz clock Internal logic levels: 0V = false (zero), +5V = true (one) Memory data logic levels: 0V = true (one), +5V = false (zero) I/O bus logic levels: +3 = false (zero), 0V = true (one)
Input/Output	Programmed I/O operations: external control, program sense, data transfer in, and data transfer out Automatic data transfers: direct memory access (DMA) with transfer rates over 382,720 words per second Interrupt system: allows computer options and peripherals to interrupt CPU operations
Standard Features	Multiply/divide and extended addressing: simplifies the programming of arithmetic and addressing operations Real-time clock: user-selected variable time base for time and event accumulation Power failure/restart: protects a program in progress during power failures

VARIAN 620/L-100 SPECIFICATIONS

Computer Option	Bootstrap protection: protects the memory addresses containing the bootstrap loader routine and the binary load/dump program
I/O Options	Priority interrupt module: establishes and implements interrupt priorities for peripherals Buffer interlace controller: permits direct access to memory for block data transfers
Input Voltage	105 to 125V ac, or 210 to 250V ac, at 50 or 60 Hz
Input Current	Power supply requires 5 amperes at 115V, and 3 amperes at 230V
Dimensions	Mainframe and expansion chassis: 10.5 inches (26.6 cm) high, 13 inches (32.9 cm) deep, and 19 inches (48.1 cm) wide Power supply: 10.5 inches (26.6 cm) high, 7.5 inches (18.9 cm) deep, and 17.75 inches (44.9 cm) wide
Weight	Mainframe and expansion chassis: approximately 35 pounds (15.9 kg) without circuit cards Power supply: approximately 36 pounds (16.3 kg)
Temperature	Operating: 0 to 50 degrees C Storage: -20 to 70 degrees C
Humidity	Operating: to 90 percent without condensation Storage: to 95 percent without condensation
Vibration	3 to 10 Hz at 1g force or 0.25 double amplitude, whichever is less; exponentially raised frequency from 3 to 10 Hz and back to 3 Hz for 10 minutes, three complete cycles; applies to all three principal axes
Shock	4g for 11 milliseconds, essentially sine shock waveform (all three principal axes, both directions in each axis)

VARIAN 620/L-100 SPECIFICATIONS

Software

Symbolic Assembler	Modular two-pass assemblers: 4K version includes 17 basic pseudo-operations; 8K version includes over 30 pseudo-operations; MR macro version operates under the 620 Master Operating System (MOS)
BLD II	Binary load/dump program that allows the loading of object programs and the punching of the binary contents of memory for reloading
Subroutines	Complete library of basic mathematical and utility subroutines; provides for the addition of special-purpose routines
FORTRAN	Modular one-pass compiler; subset of ANSI FORTRAN for 8K of memory
BASIC	An easy-to-use programming language for business and scientific applications that permits an inexperienced operator to program the system with only a few hours training
MOS	A master operating system that provides for automatic batch-processing in as little as 8K of memory
RPG IV	An optional report program generator system for business applications; produces reports, financial statements, sales records, and other commercial documents
AID II	Program analysis package that assists programmers in operating the computer and debugging other programs; includes basic operational executive subroutines
EDIT	Program modification package that allows on-line correction of symbolic source programs
MAINTAIN II	Software package that provides efficient off-line verification of CPU and peripheral operation and assists in isolating and correcting suspected faults

SECTION 30 - CENTRAL PROCESSING UNIT

The Varian 620/L-100 computer is organized in three major functional sections:

- The Central Processing Unit (CPU)
- The Memory (section 31)
- The Input/Output (I/O) System (section 7)

Figure 30-1 illustrates the functional sections of the CPU and their interaction with memory and the I/O system.

The CPU can be grouped, for descriptive purposes, into five functional sections: the control section, the arithmetic/logic section, operation registers, auxiliary register, and internal buses.

Control Section

The control section generates the timing and control signals for all computer operations. The major elements in this section are the instruction (U) register, the timing and decoding logic, and the shift control logic.

The U register receives each 16-bit instruction from memory through the W bus and holds the instruction during its execution.

The control fields of the instruction word are routed from the U register to the timing and decoding logic, where they are decoded to determine the signal levels required to perform the operations specified by the instruction.

The address field of the instruction word held in the U register is used for addressing operations. The information contained in this field is then routed to the arithmetic/logic section.

Timing logic generates the 4.211 MHz master clock from which the signals that control the sequence of computer operations are derived.

The shift control section contains the shift counter and logic to control shifting, multiplication, and division operations.

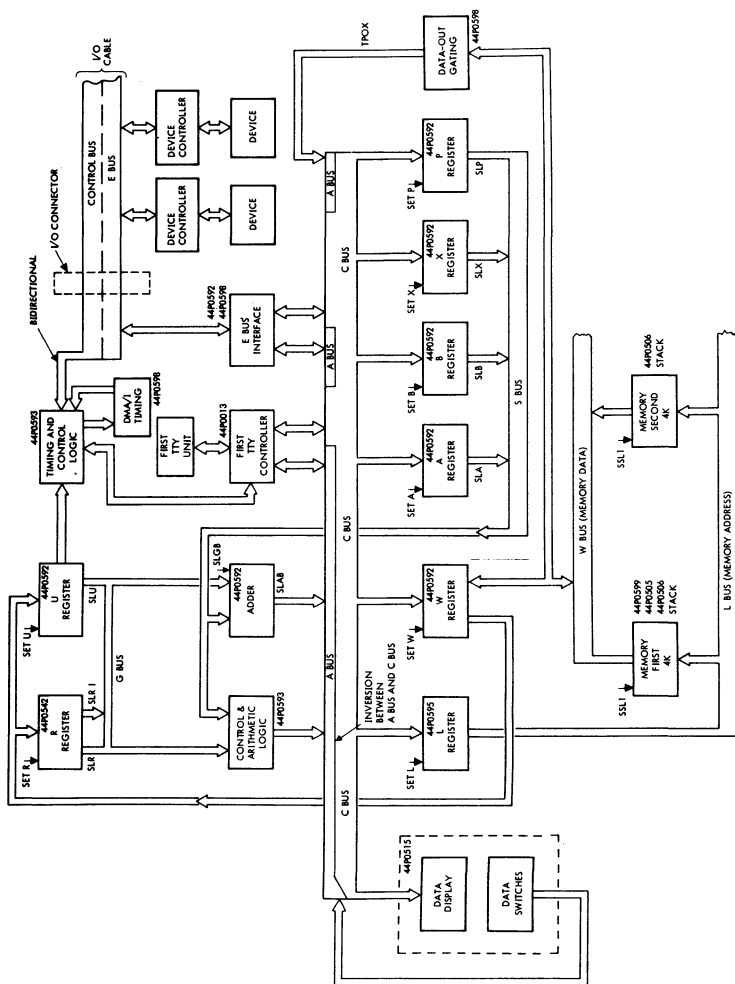
Arithmetic/Logic Section

The arithmetic/logic section comprises the operand (R) register and the arithmetic unit.

The R register receives operands from memory and holds them during instruction execution. The operand can be either data or address words. This register also permits transfers between memory and the I/O bus during the execution of the optional extended-addressing instructions.

The arithmetic unit contains gating required for arithmetic, logical, and shifting operations. Indexed and relative-addressing modifications take place in this section without adding to the instruction execution time.

The arithmetic unit also controls the gating of words from the operation registers and the I/O bus to the C bus, where they are distributed to the operation registers or to memory buffers. This facility implements various microcoded instructions (section 16).



Operation Registers

The CPU contains four operation registers, designated A, B, X, and P.

The A, B, and X registers are directly accessible to the operator. The P register is indirectly accessible through the use of the jump instructions (section 16), which modify the program sequence.

A Register

This 16-bit register is the upper-half of the accumulator. It holds the results of arithmetic and logic operations referring to operands stored in memory. During multiplication, it holds the most significant half of the double-length product. The A and B registers can also be used for I/O transfers under program control.

B Register

This 16-bit register serves as an extension of the accumulator and as a second index register. Instructions that shift the contents of the A and B registers simultaneously are available.

X Register

This 16-bit register permits indexing of operand addresses without adding time to the execution of indexed instructions.

P Register

This 16-bit register holds the address of the current instruction. It is incremented before each new instruction is fetched. A full complement of instructions is available for conditional and unconditional modification of this register (section 16). The P register is also used in relative addressing.

Auxiliary Registers

The auxiliary registers are designated U, S, L, W, and R. None are directly accessible to the operator.

U Register

This 16-bit register holds the instruction being executed. The U register acts as a buffer between the control unit and memory to permit I/O operations on a memory-cycle-by-memory-cycle basis.

S Register

This five-bit register, in combination with the U register, works as a shift counter. The S register also acts as a buffer between memory and the control unit.

L Register

This 16-bit memory address register holds the address of the location in memory being accessed during memory cycles.

W Register

The W register is the 16-bit memory buffer register.

R Register

This 16-bit buffer holds the multiplicand and divisor in arithmetic operations. The R register acts as a buffer between the arithmetic unit and memory to permit I/O operations.

Internal Buses

The CPU contains five buses, designated C, S, W, L, and E.

CENTRAL PROCESSING UNIT

C Bus

This bus provides the parallel path and selection logic for routing data between the arithmetic unit, the I/O bus, the operation registers, and the memory (W) register. The register display indicators on the computer control panel are also driven from the C bus. Collection and distribution of data simultaneously from and to operation registers is facilitated by the C bus.

S Bus

This bus provides the parallel path and selection logic for routing data from the operation registers to the arithmetic unit.

W Bus

The W register is directly connected to memory through the W bus to provide paths for data in and out of memory.

L Bus

The L register is directly connected to memory through the unidirectional L bus.

E Bus

This bus is a bidirectional input/output bus. It permits data transfers between peripheral devices and the computer. The E bus is an integral part of the I/O system (section 7).

Information Transfer

All communication between the functional sections of the CPU is through the C, S, and W buses. The C and S buses are internal to the CPU. The W bus is external and bidirectional; that is, one set of lines

carries information both to and from memory. The W bus provides a direct path to memory for data transfers and, in combination with the buffer interlace controller (BIC), allows I/O operations to occur simultaneously with extended arithmetic and shifting operations.

P Register to Memory

As an instruction cycle begins, the address of the next instruction is transferred from the P to the L register. The contents of the P register are transferred through the S bus to the adder. The adder increments the address by one and transfers the incremented count to the P register. The L register then contains the address of the instruction to be fetched from memory, and the P register holds the updated address.

Memory to U Register

During the instruction cycle, the instruction address in the L register is read out on the W bus to the W register, from which it is transferred out to the U register.

U Register to Memory

For many instructions requiring an operand, the address of the operand is in the instruction word held in the U register. This operand address is transferred to the L register through gates in the arithmetic logic and the C bus. The address from the U register can be modified during the transfer to the L register as follows:

- a. **Direct Address.** No modification; bits 0 through 10 are transferred from the U register to the L register to directly address an operand in the first 2,098 memory locations.

- b. **Relative Address.** The effective operand address transferred to the L register is formed by adding bits 0 through 8 from the U register to the contents of the P register. This permits addressing a word up to 512 locations above the current program location.
- c. **Indexed Address.** The effective operand address transferred to the L register is formed by adding bits 0 through 8 from the U register to the contents of either the X register or the B register.
- d. **Indirect Address.** The word read from memory is the address of an operand rather than the operand itself.

Memory to R Register

Operands read from memory into the W register are transferred to the R register. They are stored in the R register during an arithmetic or logical operation.

For direct addressing (and for two-word addressing instructions in which the operand address is the second word), the operand address is read from memory into the W register and then transferred to the R register; it is then routed to the L register via the C bus.

Adder to Operation Register

Outputs from the adder, generated as a result of an arithmetic operation involving the R register and one of the operation registers, are transferred to an operation register via the C bus.

Operation Register to Memory

The contents of an operation register can be transferred to memory by gating those contents to the S bus and routing the word through the C bus and the W register. Note that an address cycle must precede this transfer to load the storage address in the L register.

Input to Memory

Data from the E bus are routed directly to memory through the C and W buses. A data transfer is preceded by an address transfer to load the memory address into the L register. When the transfer is controlled by an instruction, the memory address is generated as a normal operand address.

Output From Memory

Data are transferred directly from memory to the I/O cable through the W and C buses. A storage address is first transferred to the L register by an instruction.

Input to Operation Registers

Data are transferred directly to the A or B register through the E and C buses. These transfers are always controlled by an instruction designating the register to receive the information.

Output From Operation Registers

Data are transferred directly from the A or B register to the I/O cable through the S, C, and E buses. These transfers are controlled by an instruction that connects the selected register to the S bus.

CENTRAL PROCESSING UNIT

Register to Register

The contents of an operation register can be used to replace or modify the contents of any register. The process of incrementing and restoring the contents of the P register is described above. The contents of the A, B, and X registers can be transferred, incremented, complemented, or decremented. The overflow indicator can be set and reset. These operations are implemented by gating the register contents to the S bus, processing them in the adder, and returning the result via the C bus. Note that shifting occurs in this data path. The contents of the selected register are shifted to the left or right as they are gated from the arithmetic/logic gates to the C bus. Note that all register modification instructions (section 16) use this data path.

Instruction Field Decoding

The operation code and mode (M) fields (sections 14 and 15) of the instruction word stored in the U register are decoded to provide static control levels used throughout the execution of the instruction.

Operation Code

The instruction's operation code has three functional categories: class, set, and group.

- a. Class designates one of three types of instructions: one-word addressing, one-word nonaddressing or two-word, and I/O.
- b. Set decodes simplify gating requirements for the execution of one-word addressing instructions. Timing spec-

ifications select the appropriate phase for executing the instruction.

- c. Group decoding is any arbitrary designation to describe the gating of computer operations according to the desired function. One of the group terms is true for all one-word addressing instructions.

M Field

The M field of an instruction word specifies the addressing mode or the instruction type, according to the instruction class defined in the operation code.

Timing

The Varian 620/L-100 operates on a basic 950-nanosecond machine cycle. That is, a full memory cycle (read/restore or clear/write) occurs in each 950-nanosecond interval. All computer operations take place within some multiple of this basic timing period.

During a full-cycle memory operation, suboperation timing is controlled by an internal 4.211 MHz master clock. The pulsewidth of this master clock is 237 nanoseconds, or one-fourth of the basic 950-nanosecond machine cycle; this permits the execution of various suboperations during the memory cycle. Note that the first half-cycle (475 nanoseconds) of the period is used to access a word (read) or to load zeros into an address in memory (clear). The second half-cycle is used to reload a word (restore) or to write a new word (write) into the address (section 31).

System Clocks

The clock signals that control the timing of computer operations are listed in table 30-1 and their waveforms are illustrated in figure 30-2.

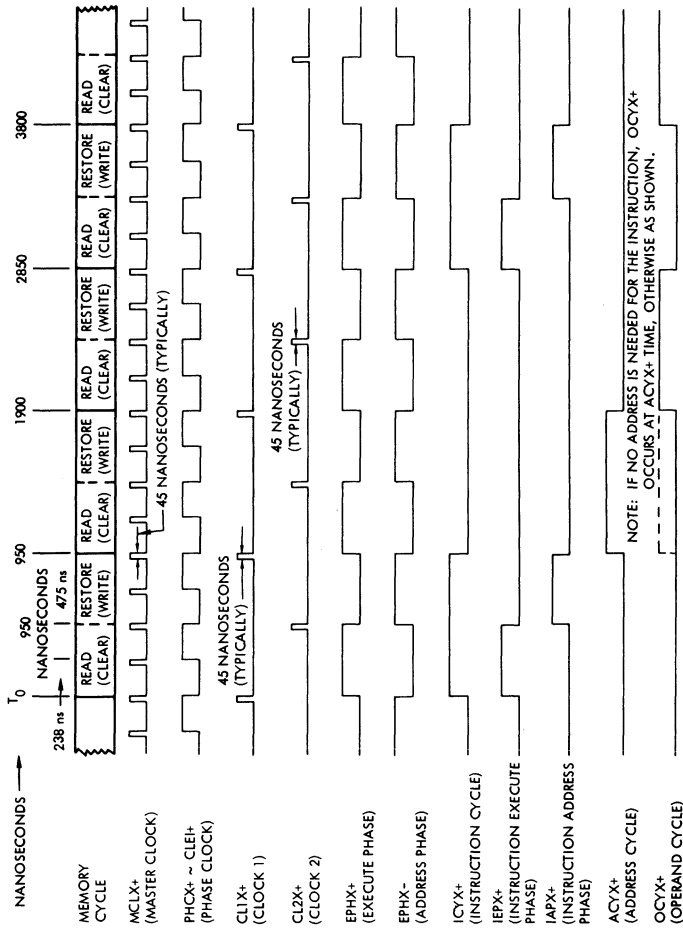


Figure 30-2. Basic Clock Waveforms

Table 30-1. Varian 620/L-100 System Clocks

Signal	Mnemonic	Description
Master Clock	MCLX +	Crystal-controlled 4.211 MHz timing signal for the entire system
Phase Clock	PHCX +	The 2.105 MHz timing signal (counted down and synchronized with MCLX +) used to time the basic address and execution phases of computer operations
Address Phase	EPHX -	Basic timing phase that corresponds to the memory restore or write half-cycle; instruction and operand addresses are transferred to memory during this period
Execution Phase	EPHX +	Basic timing phase that corresponds to the memory read or clear half-cycle; data are transferred to and from memory and instructions are executed during this period
Clock 1	CL1X +	Signal that initiates a memory cycle and operations that are synchronized with the start of the memory cycle
Clock 2	CL2X +	Signal that initiates operations that are synchronized with the start of a memory write or restore half-cycle

A memory cycle in the Varian 620/L-100 comprises two phases:

- a. Fetching an instruction from memory
- b. Executing the fetched instruction

Clock Modifiers

The memory-cycle phases are modified by certain instructions or by signals received from devices external to the computer. The conditions under which the clock periods are modified are:

Shift

During the shifting of words contained in the A and B registers, the execution phase is extended by the number of master clock periods (237.5 nanoseconds) equal to the number of shifts specified.

Interrupt

When an external interrupt is requested, the address phase is extended 475 nanoseconds to accommodate delays in receiving the interrupt address from the external device.

Trap

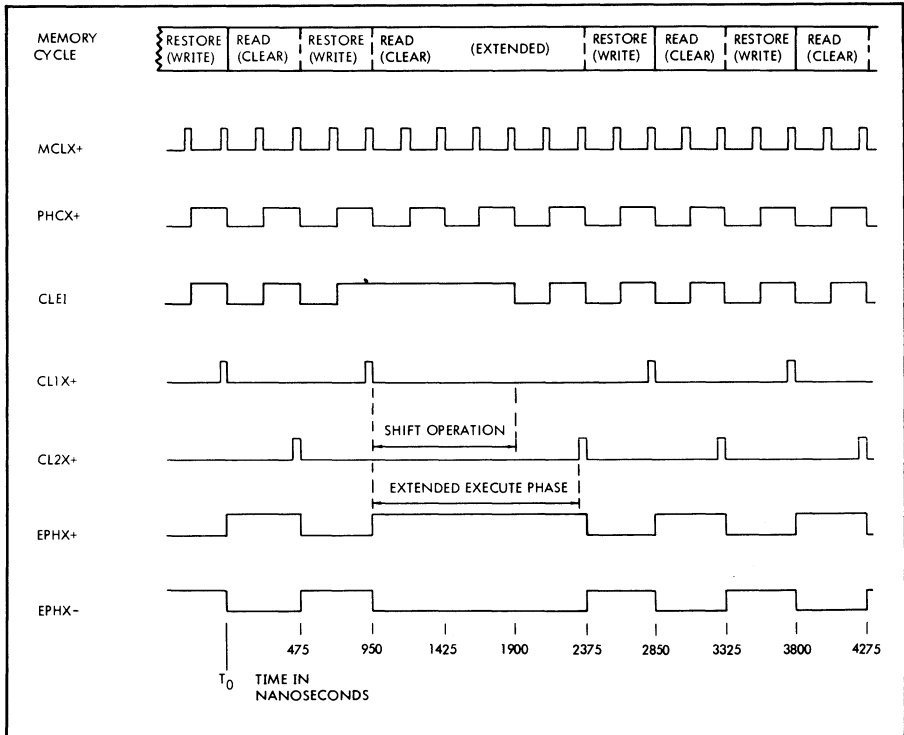
When a BIC requests a transfer to or from memory, the address phase is extended 1.66 microseconds to permit the execution of the trapping sequence (i.e., the routing of the address and data from the external device).

Halt

On a halt instruction, clock CL1X+ and CL2X+ prevent any further operations until the STEP or RUN switches are pressed to resume program execution.

Modification of the execution phase of an instruction is illustrated in figure 30-3.

The illustration is typical of a shift instruction (section 16). At time 0, the instruction



VT12-0384

Figure 30-3. Example of a Modified Clock Sequence

has been fetched from memory. Starting at time 475, the instruction is executed. However, the normal 237.5-nanosecond execution phase is extended 237.5 nanoseconds for each shift (six are illustrated). Note that CL1X+ and CL2X+ are inhibited during the extended execution phase. In a similar manner, the address phase is extended when required by the conditions defined above.

Operation Sequences

The basic clock signals generated from the 4.211 MHz master clock time three operation sequences: instruction cycle (ICYX+), operand cycle (OCYX+), and address cycle (ACYX+). All computer operations are timed by one or more of these signals.

The following paragraphs describe typical operation sequences. Variations of these sequences depend on the instruction being executed. However, a study of these fundamental operations will aid the user in understanding the timing of a specific instruction sequence.

Accessing an Operand in Memory

The simplest and most basic operation sequence is one in which a one-word, directly addressed operand is read from memory. This is typical of the load/store, arithmetic (excluding multiplication and division), and logic instructions. The timing of the suboperations of this sequence is illustrated in figure 30-4. At time 0, the instruction cycle (ICYX+) for the nth (current) instruction is initiated. Note that instruction $n - 1$ is being executed (IEPX+) while the nth instruction is being read from memory. At time 475, the instruction is transferred to the U register. During the instruction address phase

(IAPX+), when the instruction just read is being restored to memory, the operand address is generated.

Since the operand is not indirectly addressed in the illustrated case, the operand cycle (OCYX+) is initiated at time 1.8. After the operand has been read from memory and stored in the R register, the address of the next instruction ($n + 1$) is generated (normally by incrementing the P register) and transferred to the L register. This suboperation is executed while the operand is being restored to memory. The instruction cycle (ICYX+) for $n + 1$ is then initiated at time 3.6.

Note that the operation to be performed on the operand contained in the R register is executed during the IEPX+ phase of the instruction cycle for $n + 1$. This operation could be, for example, adding the operand value to the contents of the A register and storing the result in that register (ADD instruction), or simply transferring the operand to one of the operation registers (LDA, LDB, and LDX instructions, section 16).

Storing an Operand in Memory

The sequence for storing an operand in memory (STA, STB, and STX instructions, section 16) is essentially identical to that for accessing an operand, except that the specified memory address is cleared and the operand written into it. The sequence of suboperations is shown in figure 30-5.

The nth instruction is accessed and the operand address generated during the instruction cycle as described above; execution of the $n - 1$ instruction occurs during IEPX+ of the nth cycle as indicated. However, during OCYX+, the operand is transferred to memory while the referenced address is being cleared.

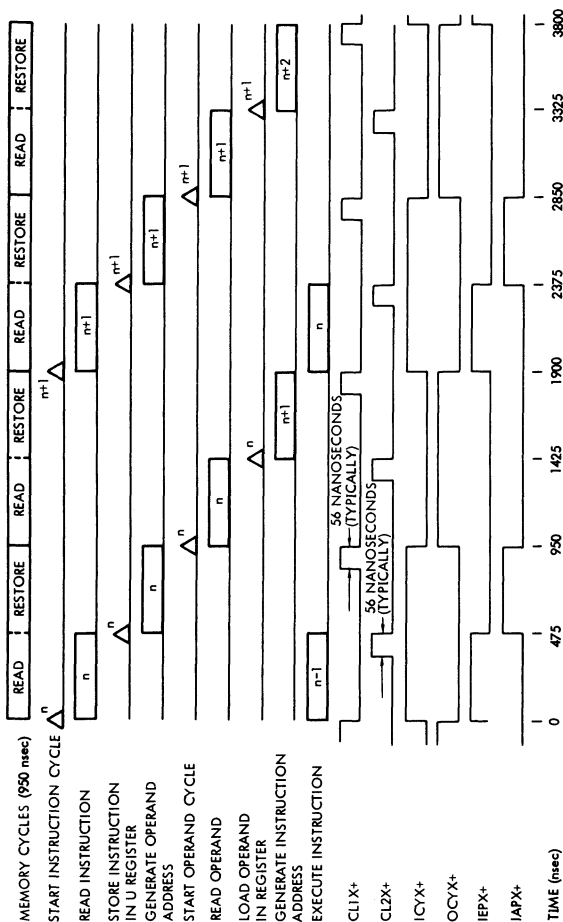


Figure 30-4. Accessing on Operand in Memory

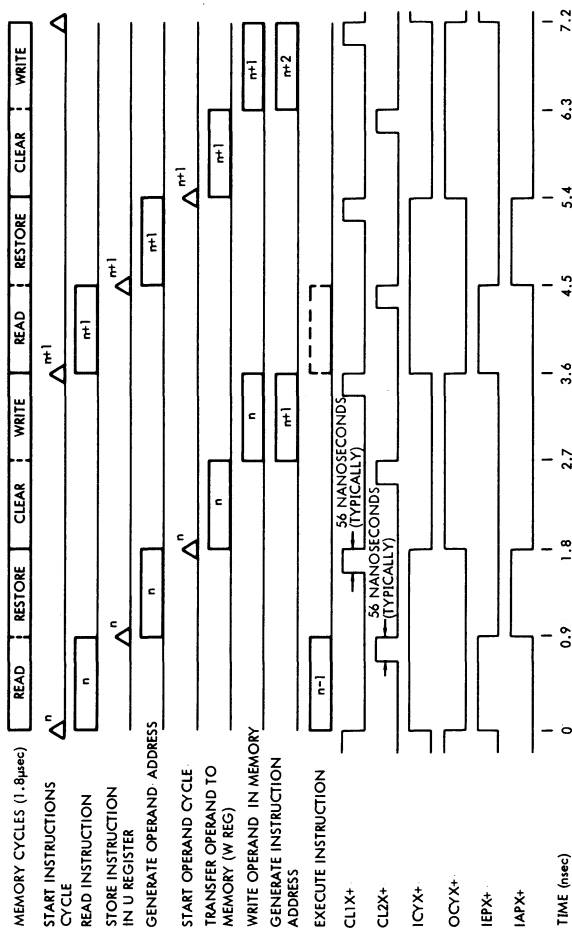


Figure 30-5. Storing an Operand in Memory

During the last half of the cycle, the operand is stored in the address just cleared. During this time, the address for the next instruction is generated. Note that there is no execution, as such, for this type of instruction (indicated by dashed lines in the illustration) because the execution has already been accomplished, in effect, by the transfer and storage of the operand in memory.

Accessing an Operand Indirectly

In this case, an address cycle (ACYX +) is required to read the indirect address word from memory before performing the operand cycle (OCYX -).

The sequence of suboperations for accessing an operand indirectly is illustrated in figure 30-6.

During the instruction cycle, the n th instruction is read from memory and stored in the U register. The previous instruction, $n - 1$, is executed during IEPX+. During the instruction address phase (IAPX+), the location of the (indirect) address word is generated. This address word is read from memory and stored in the R register.

For the case illustrated in figure 30-6, the address word contains the address of the operand. If this were not the case, another address cycle would be initiated to access a second address word, etc. The operand address is transferred to the L register during the last half of ACYX+ to locate the operand read out during the succeeding OCYX+. The address for instruction $n + 1$ is generated and instruction n is executed, completing the sequence.

Standard Features

Three standard features are available for the Varian 620/L-100:

- Multiply/Divide and Extended Addressing (M/D)
- Real-Time Clock (RTC)
- Power Failure/Restart (PF/R)

All four circuit cards can be installed in the mainframe.

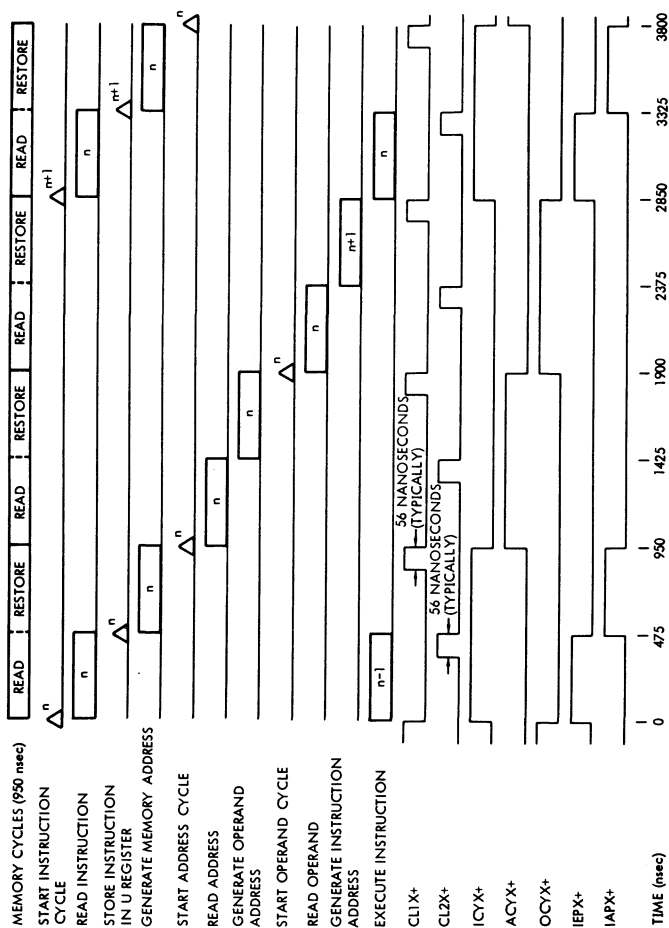
Multiply/Divide and Extended Addressing

The M/D enhances the computational capabilities of the Varian 620/L-100 computer by reducing the number of programming steps required for multiplication and division operations and by implementing the extended addressing facility.

Multiplication in the 620/L-100 system consists of successive addition of the multiplicand, with appropriate shifts to the right. Division consists of a magnitude comparison of the divisor and dividend, successive subtractions (if required), and appropriate shifts to the left of the partial quotient and partial remainder. When multiplication or division is required, a single instruction is all that is required for M/D to complete the operation.

With M/D, multiplication execution time is 9.5 microseconds, or 10 machines cycles. Execution time for division is 9.5 to 13.2 microseconds, or 10 to 14 machine cycles.

The extended addressing circuit allows addressing of all locations in memory, in



either indexed, direct, or indirect mode. Details of the extended addressing instructions are given in section 16.

M/D specifications are listed in table 30-1. Refer also to the M/D maintenance manual (document number 98 A 9902 431).

Table 30-2. M/D Specifications

Parameter	Description
Organization	Contains multiply/divide and shift control logic, and extended addressing control logic
Multiplication Algorithm	$A + B (B \cdot R) = A, B$ where A = initial A register contents B = multiplier (in B register) R = multiplicand (in memory) A, B = product (in A and B registers; most significant half in A, least significant half in B)
Multiplication Capability	Maximum multiplier: 32,767 or -32,768 Maximum multiplicand: 32,767 or -32,768 Maximum product: 1,073,741,824
Division Algorithm	$(A, B) / R = B + A$ where A, B = dividend (in A and B registers; most significant half in A, least significant half in B) R = divisor (in memory) B = quotient (in B register) A = remainder (in A register)
Division Capability	Maximum divisor: 32,767 or -32,768 Maximum dividend: 1,073,741,824 Maximum quotient: 32,767 or -32,768
Extended Addressing Modes	Relative to P (contents of second word plus P register contents plus one) Indexed with X (contents of second word plus X register contents) Indexed with B (contents of second word plus B register contents) Direct (second word is a direct address if bit 15 is zero) Indirect (second word is an indirect address if bit 15 is one)

Table 30-2. M/D Specifications (continued)

Parameter	Description
Number of Instructions	One multiplication, one division, and 14 extended addressing instructions
Logic Levels	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to 0.5V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 CM) etched-circuit card
Interconnection	Plugs into the 620/L-100 backplane
Input Power	+5V dc
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Memory Protection (Option)

The MP (Model 620/L-05) implements the designation of protected and unprotected areas of memory and prevents unauthorized entry into and modification of protected areas by programs operating in unprotected areas.

Functional Description

The MP divides computer memory into blocks (or segments) of 512 consecutive words. Under program control, these segments are selectively designated protected. Segments not specifically designated protected are, therefore, unprotected. The MP stores the protected/unprotected status of the segments in four 16-bit mask registers. These registers can store the status of up to 64 memory segments, i.e., up to 32K words.

Using the stored memory segment status, the MP monitors the address of the current instruction and that of the following instruction and the memory location specified by the effective address to determine error conditions.

When a program is operating from an unprotected segment, the MP detects the following operations as errors:

- Program overflow into a protected area
- Writing in a protected area
- Jumping to a protected area
- Executing an I/O instruction in an unprotected area
- Executing a halt instruction in an unprotected area

If these operations are attempted, the program aborts and jumps to a preassigned address in memory, from which the computer is directed to a user-written service subroutine.

Physical Description

The MP is on one 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card, which is installed in the mainframe. Interface with

the CPU is through the backplane wiring. Refer to the MP maintenance manual for details of interconnection and interfacing (document number 98 A 9902 491).

Specifications

The physical, electrical, and functional specifications of the MP are listed in table 30-2, and table 30-3 lists the MP instructions.

Table 30-3. MP Specifications

Parameter	Description
Organization	Contains an A bus receiver, device address decoder, function control logic, mask registers, segment address register, instruction address register, protected address detector, error detector, control logic, interrupt request logic, and A bus driver
Control Capability	Detects halt, overflow, I/O, write, and jump errors in up to sixty-four 512-word memory segments
I/O Capability	Six external control and eight transfer instructions
System Priority	Always assigned the highest priority in the 620/L-100 priority chain
Standard Device Address	045
Interrupt Addresses	Write error: 0120-0121 Jump error: 0124-0125 I/O error: 0130-0131 Overflow: 0134,0135
Logic Levels	Positive logic: True: +2.4 to +2.25V dc False: 0.0 to +0.45V dc Negative logic: True: 0.0 to +0.45V dc False: +2.4 to +5.25V dc

CENTRAL PROCESSING UNIT

Table 30-3. MP Specifications (continued)

Parameter	Description
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Plugs into the 620/L-100 backplane
Input Power	+5V dc at 2.5 amperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 30-4. MP Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 045	100045	Select mask register 0
EXC 0145	100145	Select mask register 1
EXC 0245	100245	Select mask register 2
EXC 0345	100345	Select mask register 3
EXC 0645	100645	Enable MP
EXC 0745	100745	Disable MP
Transfer		
OAR 0145	103145	Transfer A register contents to selected mask register
OBR 0245	103425	Transfer B register contents to selected mask register
OME 045	103045	Transfer memory contents to selected mask register

Real-Time Clock

The RTC provides flexible time orientation that can be used for time-of-day and event accumulation and as an interval time.

Functional Description

The RTC periodically interrupts the stored program to initiate user-written service subroutines that accomplish the desired and overflow. The first is a time-base

real-time functions. Two types of interrupts are generated: incrementation (interval) signal that increments a specific memory address when recognized by the computer. The second interrupt occurs when the incremented address reaches a predetermined count.

The RTC interrupt rate is determined by the frequency of the internal oscillator, or an external time-base oscillator. The internal oscillator generates a predetermined frequency within the range of 400 Hz to 80 kHz. This user-selected frequency depends on capacitor and resistor values in the oscillator circuitry; it can be changed by replacing one or more of a group of eight components.

Physical Description

The RTC is on one 7-3/4-by-12-inch (19.7 x 30.3 cm) etched circuit card, which is installed in slot 16 of the mainframe. If the PF/R is also included in the system, it is located on the same card. Interface with the CPU is through the backplane wiring. Refer to the RTC maintenance manual for details of interconnection and interfacing (document number 98 A 9902 451).

Specifications

The physical, electrical, and functional specifications of the RTC are listed in table 30-4, and table 30-5 lists the RTC instructions.

Table 30-5. RTC Specifications

Parameter	Description
Organization	Contains an address decoder, internal time-base oscillator, divide-by-eight frequency counter, overflow detector, interrupt memory register, interrupt control logic, and drivers and receivers
Control Capability	Initiates incrementation (interval) and overflow interrupts
Frequency Range	400 Hz to 80 kHz (user-selected)
I/O Capability	Four external control instructions
System Priority	Determined by location in the 620/L-100 priority chain (normally second to the PF/R)
Standard Device Address	047
Interrupt Addresses	Interval: 044-045 Overflow: 046-047

Table 30-5. RTC Specifications (continued)

Parameter	Description
Logic Levels	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc Negative logic: True: 0.0 to +0.45V dc False: +2.8 to +3.6V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Plugs into the 620/L-100 backplane
Input Power	+5V dc and +12V dc
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 30-6. RTC Instructions

Mnemonic	Octal Code	Description
External Control		
EXC 0147	100147	Enable RTC (both incrementation and overflow interrupts)
EXC 0447	100447	Initialize RTC (inhibits interrupts and resets interrupt register and divide-by-eight counter)
EXC 0247	100247	Inhibit overflow interrupts
EXC 0347	100347	Enable incrementation interrupts and inhibit overflow interrupts

Power Failure/Restart**Functional Description**

The PF/R (Model 620/L-14) provides an orderly shutdown in case of power failure or turn-off and restarts the interrupted program when power is restored.

Power input to the computer is indirectly monitored by the PF/R. A power failure monitor voltage in the computer power supply is constantly being sensed to deter-

mine power status. If the voltage drops (because of power failure or turn-off), the PF/R initiates an interrupt. The CPU then executes a user-written service subroutine (SAVE) that places the contents of the volatile registers (A, B, X, P, and overflow) in memory. Execution of the stored program stops, memory is disabled, and the system is reset. This power-down subroutine cannot be interrupted by lower-priority options or peripheral controllers.

When power is restored, the PF/R enables the memory. The CPU executes a user-written service subroutine (RESTORE) that reloads the contents of the volatile registers, and the system resumes execution of the program in progress at the time of the interrupt.

If power fails while the computer is in step mode, memory is protected, but the contents of the volatile registers are lost.

Physical Description

The PF/R is on one 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card, which is installed in slot 16 of the mainframe. If the RTC is also included in the system, it is located on the same card. Interface with the CPU is through the backplane wiring. Refer to the PF/R maintenance manual for details of interconnection and interfacing (document number 98 A 9902 442).

Specifications

The physical, electrical, and functional specifications for the PF/R are listed in table 30-6.

Table 30-7. PF/R Specifications

Parameter	Description
Organization	Contains a control sequencer, interrupt request logic, and control logic
Control Capability	Initiates power failure and power restart interrupts
System Priority	Determined by location in the 620/L-100 priority chain (normally second to the MP)
Standard Device Address	47
Interrupt Addresses	Power failure (SAVE): 040-041 Power restart (RESTORE): 042-043
Timing	Power failure detection: less than one cycle of the ac input voltage after failure or turn-off Power-down: less than 500 microseconds after failure detection Power-up: less than 1.1 seconds after power restoration

Table 30-7. PF/R Specifications (continued)

Parameter	Description
Logic Levels	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc Negative logic: True: 0.0 to +0.5V dc False: +2.8 to +3.6V dc
Size	One 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit card
Interconnection	Plugs into the 620/L-100 backplane
Input Power	+5V dc, +12V dc, and -12V dc
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

SECTION 31 - MEMORY

The Varian 620/L-100 memory is a parallel, random-access, three-wire/three-dimensional, magnetic-core memory. It provides program and data storage for the Varian 620/L-100 computer. The basic computer system package accommodates 4,096 (4K) or 8,192 (8K) sixteen-bit words. The memory can be expanded to 32,768 (32K) words in 4K increments.

Memory Design

The basic information storage element of the memory is the magnetic core. The core is a toroid of ferrite that can be magnetized in two discrete directions representing a binary one or zero. The core is magnetized by a current passing through it. The direction of current flow through the core determines the direction of magnetization, either read or write.

The Varian 620/L-100 memory uses the coincident-current technique for core magnetization. Two perpendicular wires (X drive and Y drive) pass through each core. During memory operations, the current on each drive wire is approximately one-half the current necessary to magnetize a core; thus, only the core at the intersection of two activated drive wires is magnetized.

Memory Operations

A memory full-cycle consists of a reading sequence followed by a writing sequence. The full-cycles are:

- a. **Clear/write:** a word is transferred from the CPU to memory

- b. **Read/restore:** a word is transferred from memory to the CPU

Both full-cycle operations require 950 nanoseconds, and memory access time is 425 nanoseconds.

A clear/write operation loads the memory. The clear half-cycle of the operation resets the addressed cores to zero; the write sequence then loads the data in the cores.

A read/restore operation reads information from the memory. The read half-cycle of the operation unloads the addressed data word from memory; the restore sequence immediately reloads (writes) the word in the same location.

Half-cycle operation is required for trap-in functions of the I/O system (section 7). Memory-CPU operations can be halted after the first half-cycle (either clear or read) of the full-cycle sequence so that external system devices have direct access to data in memory.

Circuit Cards

The Varian 620/L-100 memory circuits are contained on the following circuit card types:

- a. **Memory timing control (MTC) card:** Generates control signals for the total memory system.
- b. **Driver/sink switch (DSS) card:** Provides X and Y drive for up to 8K of memory.
- c. **Memory stack (MS) card:** Consists of a diode-decoding matrix and a

SECTION 31 MEMORY

planar magnetic core array composed of sixteen 4K mats (a mat represents one bit of the 16-bit word). Each mat has 64 rows (X lines) and 64 columns (Y lines) of magnetic cores and an individual sense/inhibit line. Two diodes per each X/Y line select the line(s) to be activated.

- d. *Sense/inhibit (SI) card:* Contains 16 pairs of sense amplifiers and inhibit drivers, each pair being associated with one bit of data. The sense amplifier compares the sense line voltage with a predetermined threshold voltage to determine the state of a core, and the inhibit driver provides a current to reset a data bit to zero.

Figure 31-1 illustrates the functional circuit blocks of the memory in relation to the circuit cards.

Master Memory Assembly

The basic Varian 620/L-100 memory assembly is either 4K or 8K, designated the master memory, and installed in the computer mainframe.

The 4K master memory consists of: One MTC card, one DSS card, one MS card, and one SI card.

The 8K master memory consists of: One MTC card, one DSS card, two MS cards, and two SI cards.

Memory Expansion

Memory is expanded from the basic 4K or 8K master with either 4K or 8K slave assemblies in the memory expansion chassis. The maximum 32K system requires only one such chassis.

The slave memory assemblies correspond to the master assemblies, except that they have no MTC card.

The addition of slave memory assemblies to the computer system requires that a special memory buffer card be installed in the expansion chassis to buffer W and L bus signals to and from the chassis.

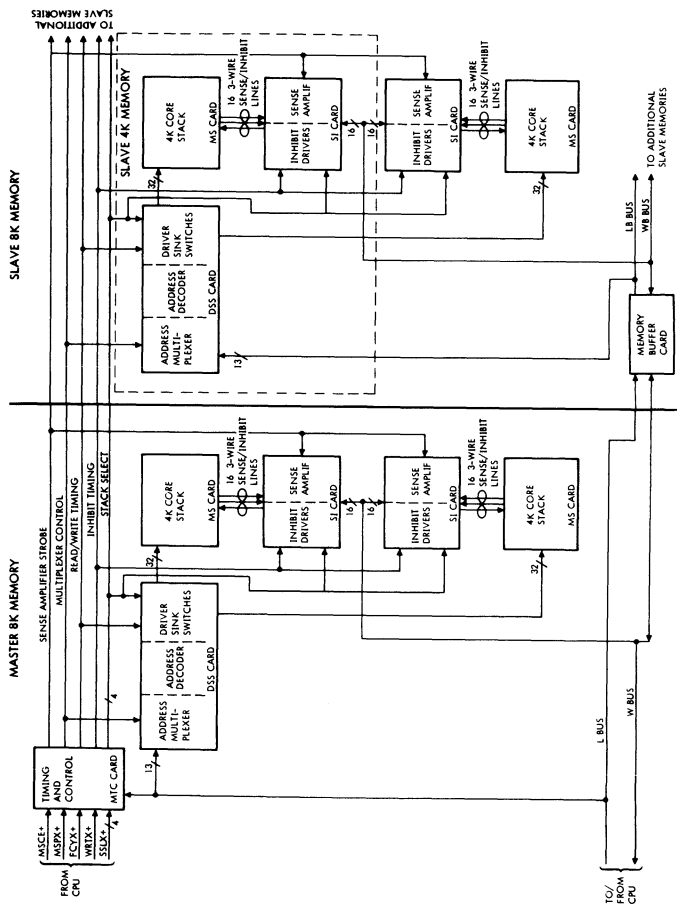
Interfacing and Timing

Addresses are sent to memory through the L bus, and data are transferred to or from memory through the W bus.

The CPU requests access to memory by raising a memory start pulse (figures 31-2 and 31-3). The memory cycle begins after a delay for decoding. The memory performs a clear/write or read/restore operation, depending on the level of the write/read control line (WRTX+). For read/restore, data are available on the W bus 420 nanoseconds after the leading edge of the memory start pulse. During the 550- to 950-nanosecond interval after the leading edge of the pulse, data on the W bus are written into memory, completing the write portion of the cycle.

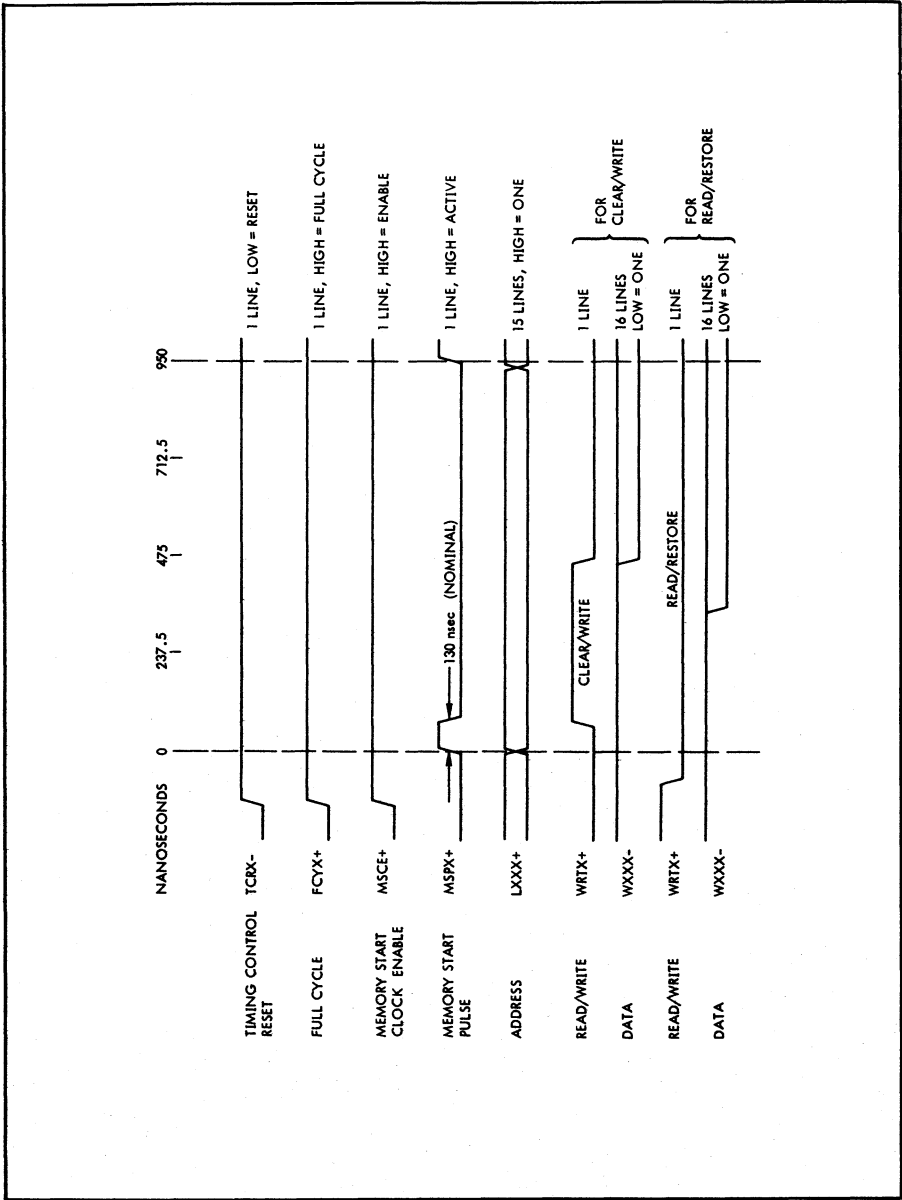
The memory can execute half-cycle or full-cycle operations, depending on the level of the cycle control line (FCYX+) from the CPU. FCYX+ is low for a half-cycle, suspending memory-CPU operations after completion of the read half-cycle. Memory then waits for a second memory start pulse before performing the write half-cycle of the operation.

To protect the contents of memory during power turn-on/off, the Varian 620/L-100 CPU logic includes power/failure restart (PF/R).



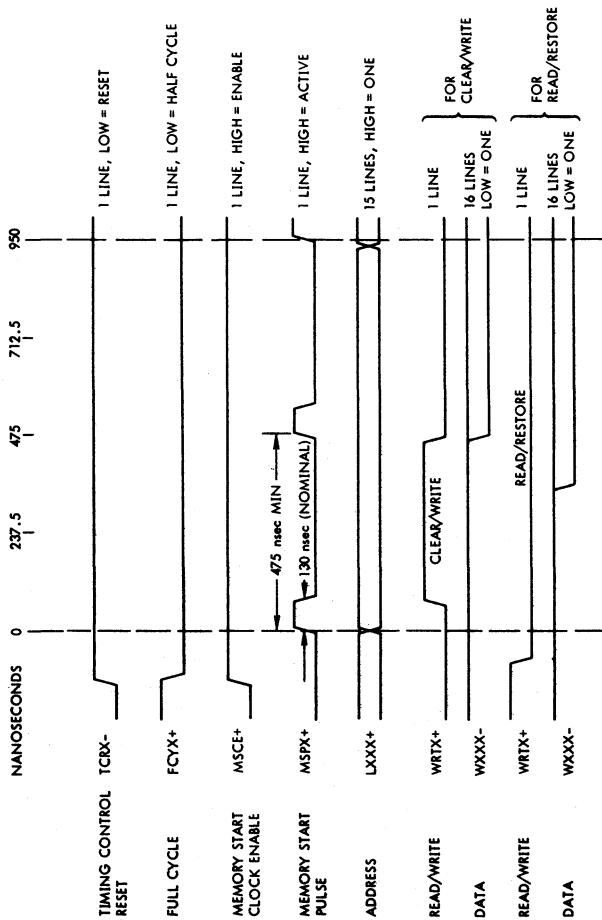
VT13-0274

Figure 31-1. Memory Block Diagram



VT11-1385A

Figure 31-2. Memory Full-Cycle Timing



V711-13864

Figure 31-3. Memory Half-Cycle Timing

Wrap-Around Addressing

Wrap-around addressing is available (upon request) as a standard feature of the Varian 620/L-100 memory. Wrap-around addressing techniques automatically prevent the CPU from trying to address data to or from nonexistent memory addresses. Without this capability, data read out of a nonexistent address result in zero, and data written are lost.

Wrap-around addressing is implemented with jumper connections on the CPU backplane.

In a computer with a 4K memory (symbolically, addresses 0 through 4,095) wired for wrap-around addressing, each existent memory location can be addressed directly and by seven corresponding addresses in nonexistent memory; e.g., address 0 can be addressed by 0, by 4,096, by 8,192, by 12,288, by 16,384, by 20,480, by 24,576,

and by 28,672. For 8K wrap-around, each existent location can be addressed by its own designation plus three nonexistent addresses, and for 16K, its own plus one nonexistent address.

Loader Protection

The loader protection option (section 32) prevents writing into the areas of memory that hold the bootstrap loader routine and binary load/dump program. This option is implemented by a toggle switch located behind the control panel. The loader protection circuitry is on the MTC circuit card.

Specifications

Performance specifications for the Varian 620/L memory system are listed in table 31-1.

Table 31-1. Memory Specifications

Parameter	Description
Description	Parallel, random-access, 3W/3D, magnetic core memory
Cycle Time	950 nanoseconds
Access Time	425 nanoseconds
Execution Times	Write data available from 550 to 950 nanoseconds; address available at 60 nanoseconds (settled); memory start pulsewidth, 40 nanoseconds (minimum), 150 nanoseconds (maximum)
Operating Modes	Continuous at any address, and in burst mode nonoperating time

Table 31-1. Memory Specifications (continued)

Parameter	Description
Operating Limits	X/Y current: ± 5 percent (minimum) Inhibiting current: ± 5 percent (minimum) Power supply voltages: can be biased ± 5 percent from nominal
Operating Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

SECTION 32 - COMPUTER OPTION

Bootstrap Protection

The **Bootstrap Protection (BP)** option (Model 620/L-115) protects the memory addresses containing the bootstrap loader routine and binary load/dump (BLD II) program.

BP is implemented with a toggle switch behind the computer control panel. Positioning this switch to **DISABLE** makes all of memory available for the storage of data and instructions. When the switch is set to

ENABLE, data written in the addresses reserved for the bootstrap routine and BLD II cannot be overwritten.

An attempt to write in the area protected by the BP changes the write cycle to read cycle and the computer either halts or continues cycling with or without interrupt operation. Under interrupt control, the BP transmits an interrupt request to the priority interrupt module (PIM). The writing operation causing the interrupt is changed to a reading operation, the interrupt occurs when system priority permits.

The BP specifications are listed in table 32-1.

Table 32-1. BP Specifications

Parameter	Description
Organization	Consists of control logic to detect addresses occupied by the bootstrap loader routine and BLD II, and to generate CPU and memory control signals
Modes of Operation	Operates under both program and interrupt control
Control Capability	Protects the last 0256 or 0400 addresses of a selected 4K memory increment
Implementation	Enabled/disabled with a toggle switch behind the computer control panel (lower left corner of the mainframe)
Logic Levels	Positive logic: True: +2.5 to +5.0V dc False: 0.0 to +0.5V dc
Construction	ICs mounted on a portion of the memory timing and control circuit card
Interconnection	Plugs into the 620/L-100 backplane
Input Power	+5V dc at 100 milliamperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

SECTION 33 - PHYSICAL CONFIGURATION

An outstanding feature of the Varian 620/L-100 is its compact design. A fully expanded system fits into just 21 inches (53.1 cm) of vertical 19-inch (48.1 cm) rack space.

The Varian 620/L-100 is also convenient to maintain and troubleshoot. All circuitry, other than that for the control panel and the power supply, is contained on circuit cards that plug into the mainframe or in one or more expansion chassis.

Circuit Cards

Varian 620/L-100 computer systems are assembled from the circuit cards listed in table 33-1 and the controller cards for the peripherals and I/O interfaces described in section 9.

These circuit cards are 7-3/4-by-12-inch (19.7 x 30.3 cm) etched-circuit or wired-socket types, and except for the control panel display and memory stack cards, they plug into the backplane connectors of the mainframe and expansion chassis.

System Planning

Varian 620/L-100 systems are packaged in two basic types of chassis: the mainframe and expansion. All systems also require a power supply separate from the computer chassis, but mounted in the same rack enclosure.

Space Requirements

The mainframe, expansion chassis, and power supply are in individual cabinets suitable for standard RETMA rack or table-top installation.

Table 33-1. Varian 620/L-100 Circuit Cards

Assy No.	Description	Assy No.	Description
44P0592	Processor register		
44P0595	Processor control 1		
44P0596	Processor control 2		
44P0597	Processor control 3	44P0248*	Memory protection
44P0593	Processor control 4	44P0254*	I/O buffer
44P0013	Teletype controller	44P0599	Memory timing control
44P0598	DMA and interrupt	44P0578	Memory driver/sink switch
44P0594	Multiply/divide and extended addressing	44P0506	Memory sense/inhibit and memory stack
44P0237	Power failure/restart (PF/R) and real-time clock (RTC)	44P0515	Control panel display
		44P0521	Memory buffer

*Mainframe Option

PHYSICAL CONFIGURATION

Both the mainframe and expansion chassis are 10.5 inches (26.6 cm) high, 13 inches (23 cm) deep, and 19 inches (48.1 cm) wide.

The power supply is 6 inches (15.2 cm) high, 10.2 inches (26 cm) deep, and 17.5 inches (45 cm) wide.

The 33 ASR Teletype unit with stand, which is included in most systems, is approximately 33 inches (83.5 cm) high, 19 inches (48.1 cm) deep, and 22 inches (55.77 cm) wide.

Expansion chassis can contain all-memory, memory-and-I/O, and all-I/O configurations. Chassis containing either of the first two configurations must be located directly above or below the mainframe. An all-I/O expansion chassis can be connected to the mainframe by I/O expansion cables up to 20 feet (6m) in length.

In equipment rack installations, the power supply is normally in a hinged, swing-down chassis mounted at the rear of the mainframe or expansion chassis. Any installation should provide easy access to power supply adjustment controls, test points, and switches.

Power Requirements

One power supply provides power for the CPU, 32K of memory, and all the internal standard features and options. When peripheral controllers are added to this expanded system, a second power supply is required.

The power supplies connect to a standard 115V, 60 Hz source. Regulation is not required under normal commercial power conditions.

With maximum loads, the Varian 620/L-100 power supplies draw approximately 15 amperes of ac line current.

Also available, for European installations, are power supplies for use with 230V, 50 Hz power sources.

Environmental Requirements

Ambient temperature at Varian 620/L-100 installations can vary between 0 and 50 degrees C with no adverse effects on computer operations. To extend the life expectancy of the computer, however, it is recommended that ambient temperature be maintained between 15 and 30 degrees C.

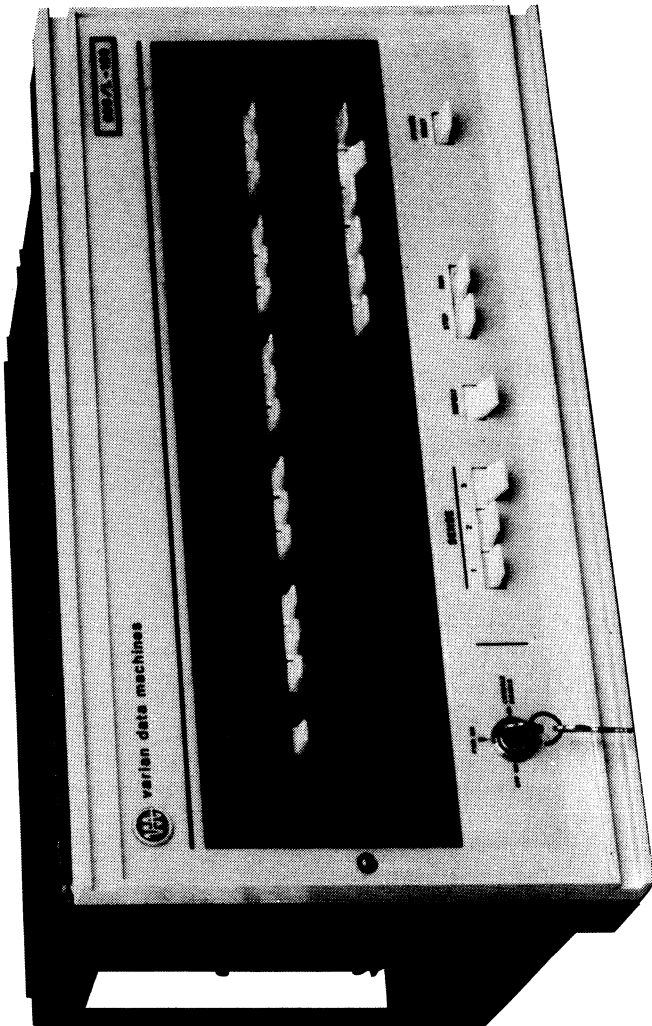
Relative humidity, without condensation, can vary between 0 and 90 percent.

Physical Description

Mainframe

The Varian 620/L-100 mainframe (figure 33-1) contains a control panel, a connector panel, and card slots for CPU, memory, computer and I/O options, and selected peripheral controllers. A fan under the mainframe provides cooling for the electronic components. Circuit cards plug into the backplane wiring board located behind the control panel.

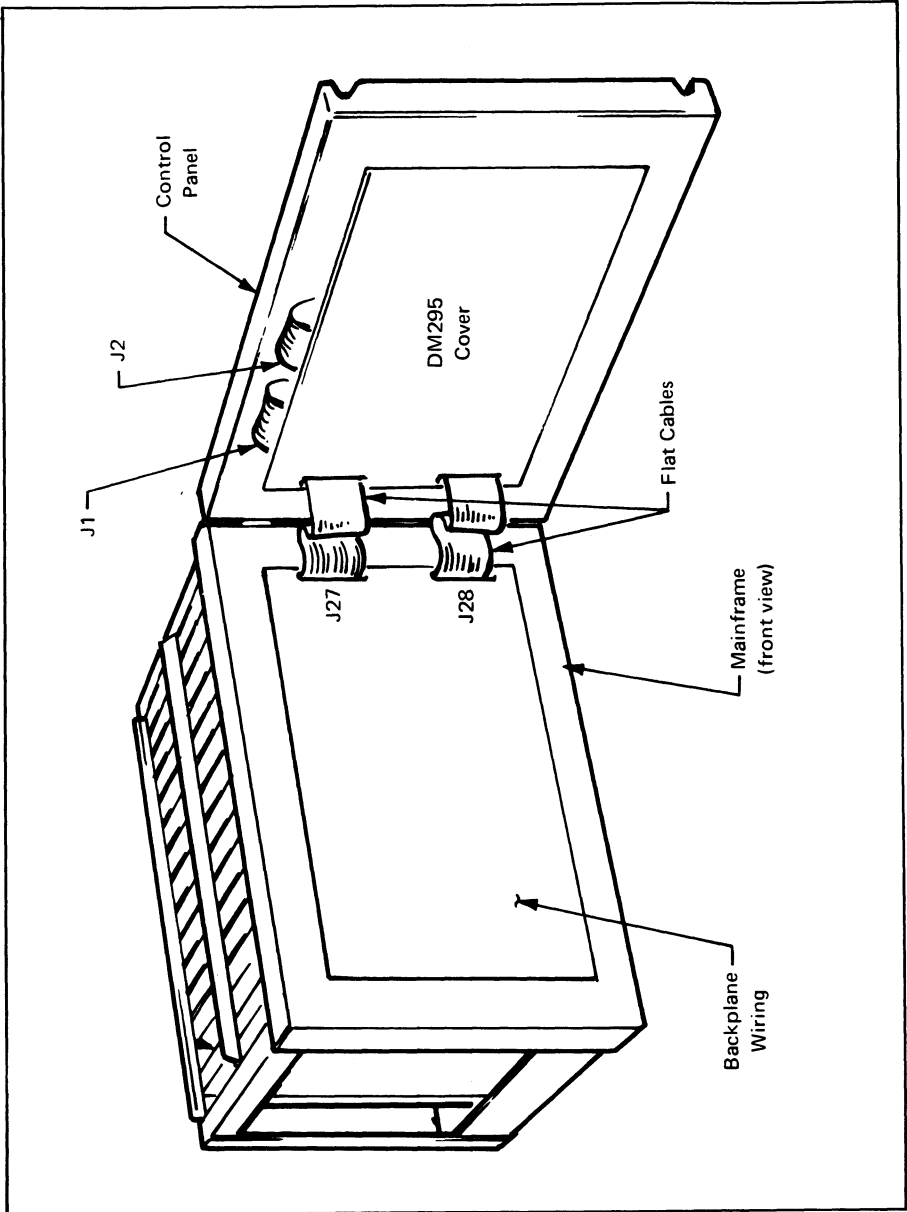
The molded-plastic control panel contains all the controls and indicators necessary for computer operation. The control panel display circuit card (44P0515) is installed behind the panel. This card contains the lamps, switches, and associated circuitry (the lamps can be replaced without soldering). The display card connects to the CPU via two flat cables plugged into connectors J27 and J28 on the backplane (figure 33-2).



VHT0-0172

Figure 33-1. Varian 620/L-100 Mainframe

PHYSICAL CONFIGURATION



VTII-1248

Figure 33-2. Control Panel Opened to Expose Backplane

The control panel is hinged to the front of the mainframe and can be opened to expose the backplane wiring by unlatching the pushbutton fastener on the left side of the mainframe.

The connector panel, described in detail in section 34, is located at the rear of the

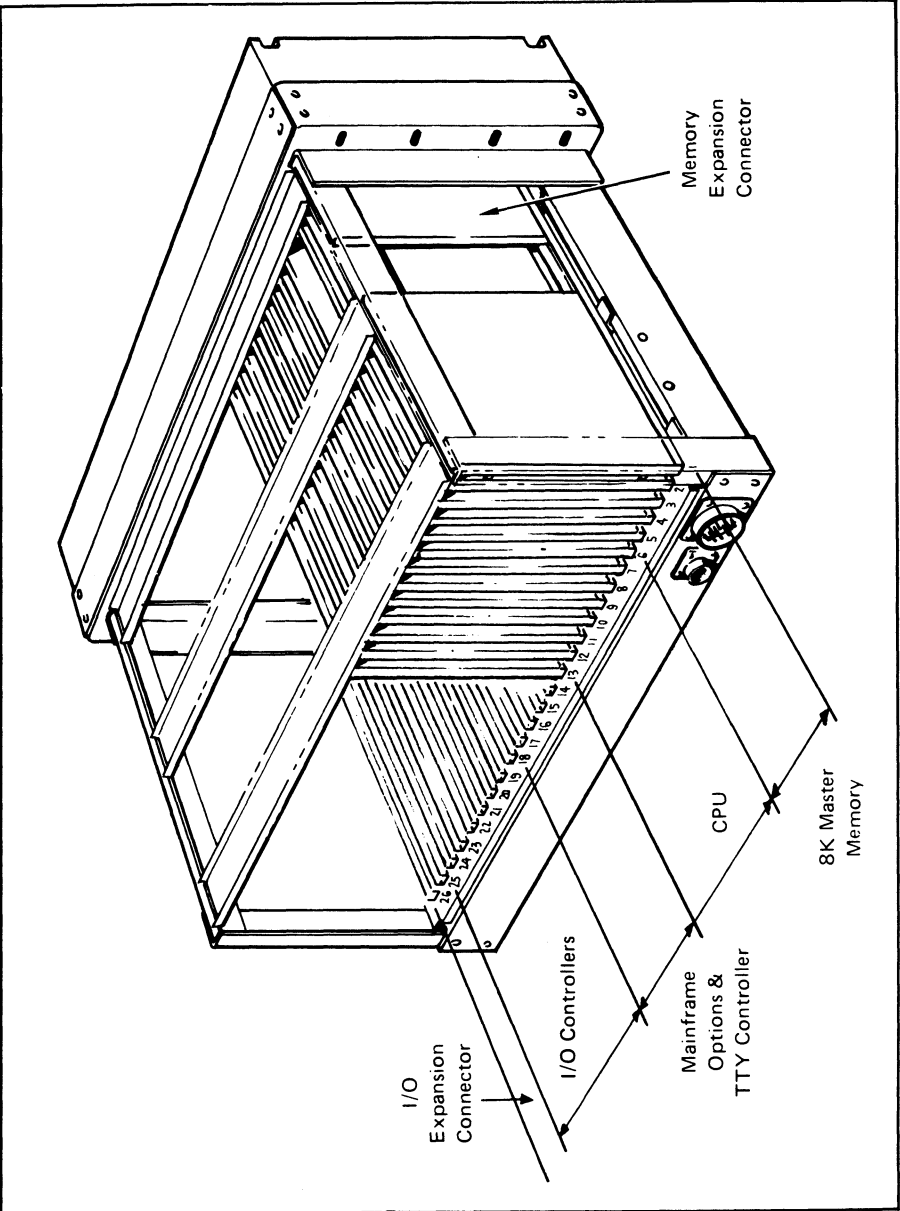
mainframe below the circuit card slots. This panel provides power and Teletype connectors.

Twenty-six circuit cards can be accommodated in the mainframe card slots. The standard card slot assignments are illustrated in figure 33-3, and listed in table 33-2.

Table 33-2. Mainframe Card Slot Assignments

Card Slot	Assy No.	Description
1*	----	Memory expansion connector and memory stack
2	44P0506	Memory sense/inhibit
3	44P0578	Memory driver/sink switch
4*	----	Memory stack
5	44P0506	Memory sense/inhibit
6	44P0599	Memory timing control
7	44P0592	Processor register
8	44P0592	Processor register
9	44P0592	Processor register
10	44P0595	Processor control 1
11	44P0596	Processor control 2
12	44P0597	Processor control 3
13	44P0593	Processor control 4
14	44P0594	Multiply/divide and extended addressing
15	44P0598	DMA and interrupt
16	44P0237	Power failure/restart and real-time clock
17	44P0013	Teletype controller
18	----	Not wired
19-25	----	Peripheral controllers
26	----	I/O expansion connector

* The memory stack cards occupy card slots 1 and 4 but do not plug into the backplane connector; they plug into a right-angle connector on the memory sense/inhibit cards.



VTII-1249

Figure 33-3. Mainframe Circuit Card Locations

Circuit cards are installed in the 122-pin backplane connectors through the rear of the mainframe with the component side of the card on the installer's left (except for the memory stack cards on which the component side is on the right).

Card slot 18 is normally not wired. It can, however, be used for special options or for peripheral controllers that require more than one card slot.

Memory expansion cable connectors are circuit cards. The circuit card connector at one end of the cable plugs into slot 1 of the mainframe; the other, into slot 2 of the expansion chassis. The same type of cable is used for I/O expansion. The I/O expansion cable plugs, at one end, into slot 26 of the mainframe, and, at the other, into slot 13 of the right half of the expansion chassis (as viewed from the front).

System interconnection details are given in section 34.

Expansion Chassis

Varian 620/L-100 expansion chassis (figure 33-4) contain a front panel, a connector panel, and card slots for memory and/or peripheral controller cards.

The front panel is blank and hinged so that it can be opened to expose the chassis wiring. (in the same manner as the front panel on the mainframe).

The connector panel is similar to that of the mainframe. It is also located at the rear of the chassis, below the card slots (section 34).

Card slots in the expansion chassis are divided into two sets of 12, each half numbered from 2 through 13 reading from

right to left when viewed from the rear of the chassis. Circuit cards are installed in the expansion chassis in the same manner as in the mainframe.

Fans in the bottom of the expansion chassis provide cooling for memory expansion modules (one for the first 8K of memory, and two for additional modules). Cooling fans are not included in an expansion chassis containing only peripheral controllers.

Memory Expansion

Memory is expanded with 4K or 8K slave memory modules installed in an expansion chassis. Card slot assignments for an all-memory expansion chassis are listed in table 33-3, and illustrated in figure 33-5. A combination memory-I/O expansion chassis is illustrated in figure 33-6.

An 8K slave memory module is identical to the 8K master in the mainframe (slots 2 through 6, table 33-2), except that the memory timing control card (44P0599) is not included.

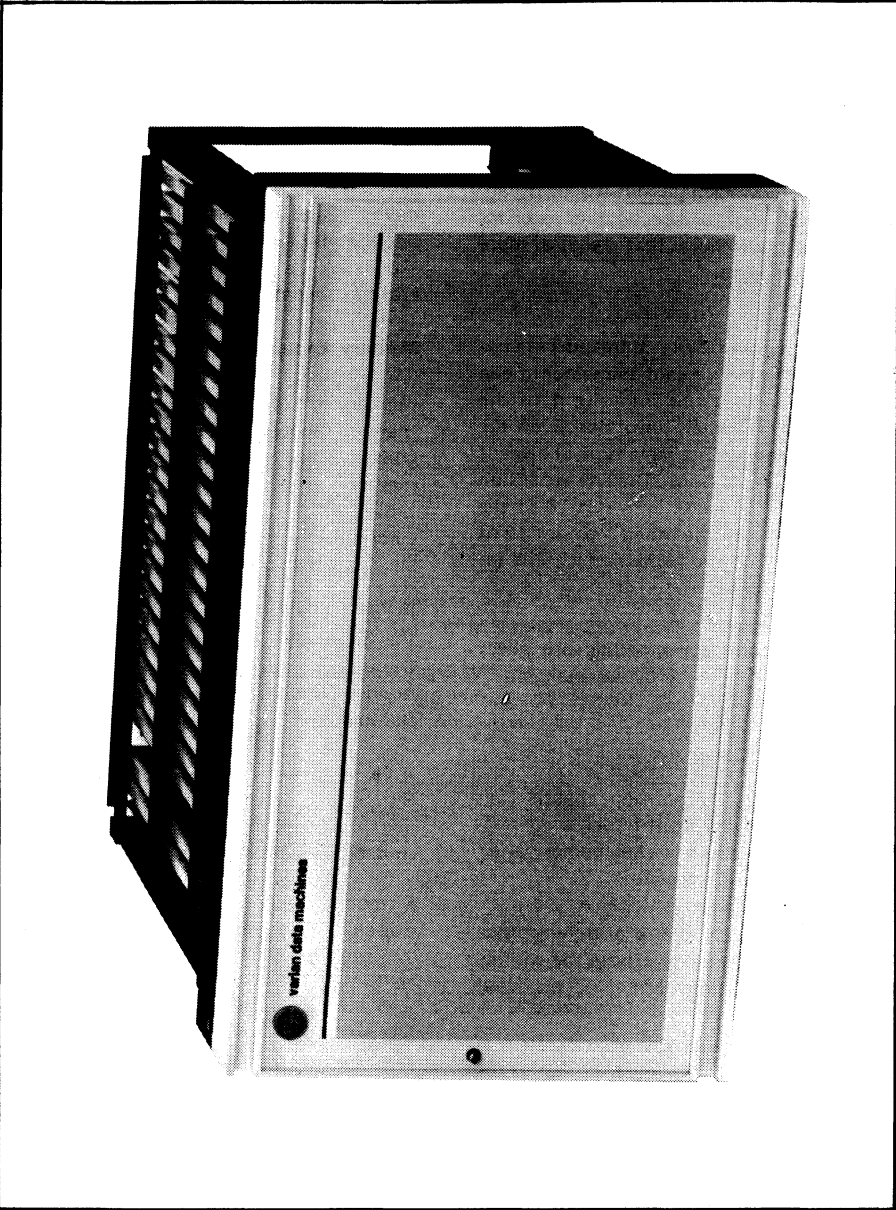
A 4K slave memory module is similar to an 8K module, except it has only one memory sense/inhibit card (44P0506) and one memory stack card.

A memory buffer card (44P0521) is required for a larger than 8K memory.

Computer systems with 12K, 16K, 20K, or 24K memories require an expansion chassis with a left-half memory expansion backplane. Larger systems require a right-half backplane also.

I/O Expansion

An all I/O expansion chassis is illustrated in figure 33-7. Peripheral controller cards require either one or three card slots.



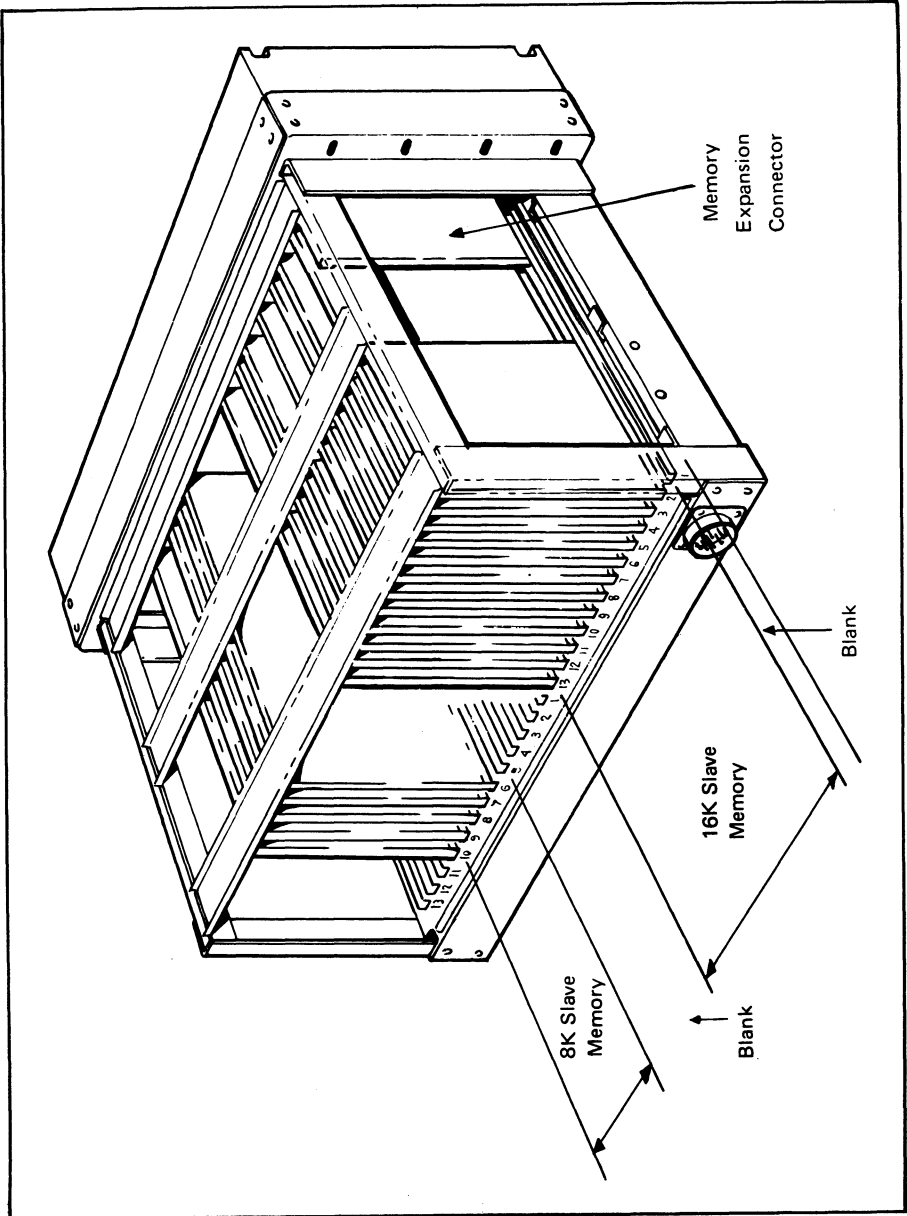
VHT0-0173

Figure 33-4. Varian 620/L-100 Expansion Chassis

Table 33-3. All-Memory Expansion Card Slot Assignments

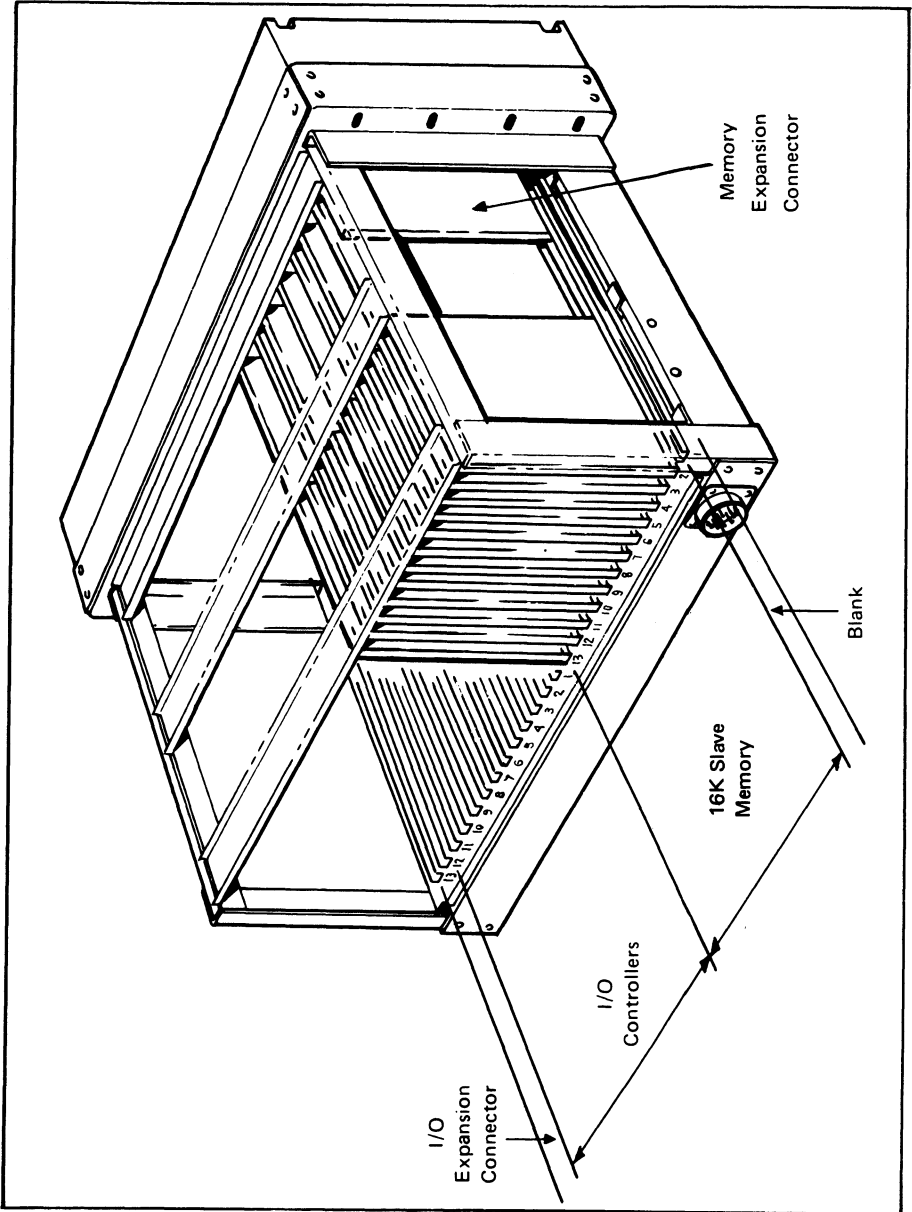
	Slot	Description		
Left-Half Memory Expansion Backplane	1	Blank		
	2	Memory expansion connector		
	3	Memory buffer		
	4*	Memory stack	1st	
	5	Memory sense/inhibit	4K	1st
	6	Memory driver/sink switch		8K
	7*	Memory stack	2nd	
	8	Memory sense/inhibit	4K	
	9*	Memory stack		
	10	Memory sense/inhibit	3rd	
	11	Memory driver/sink switch	4K	2nd
	12*	Memory stack	4th	8K
	13	Memory sense/inhibit	4K	
Right-Half Memory Expansion Backplane	1	Blank (Slots 2 through 5 not used)		
	6*	Memory stack	5th	
	7	Memory sense/inhibit	4K	
	8	Memory driver/sink switch		3rd
	9*	Memory stack	6th	8K
	10	Memory sense/inhibit (Slots 11 through 13 not used)	4K	

* The memory stack cards occupy the indicated slots, but do not plug into the connectors; they plug into right-angle connectors on the sense/inhibit cards.



VT11-1250

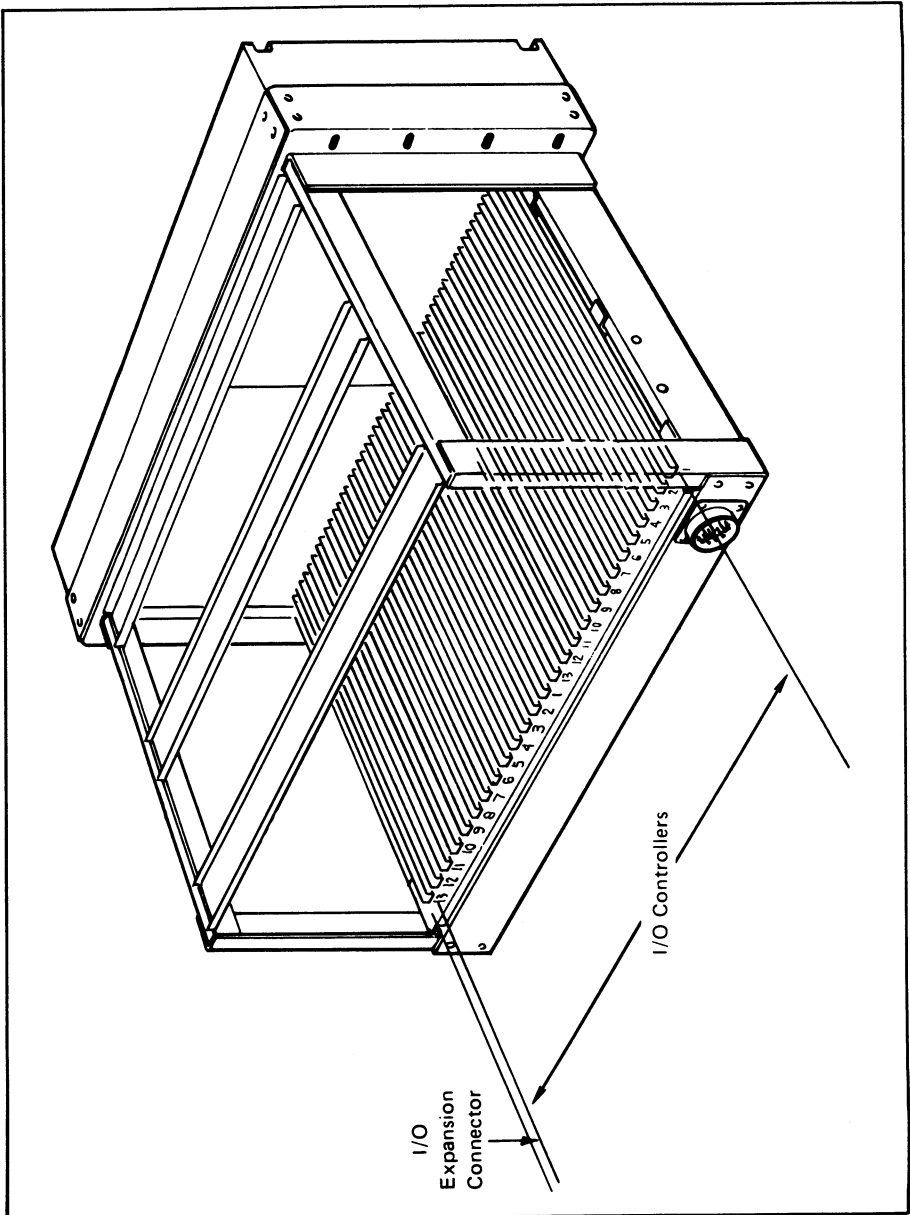
Figure 33-5. All-Memory Expansion Card Locations



VTII-1251

Figure 33-6. Memory-and-I/O Expansion Card Locations

PHYSICAL CONFIGURATION



VTII-1252

Figure 33-7. All-I/O Expansion Card Locations

An expansion chassis equipped with a right-half I/O backplane can be used to provide 12 peripheral controller card slots (slot 13 holds the I/O expansion connector).

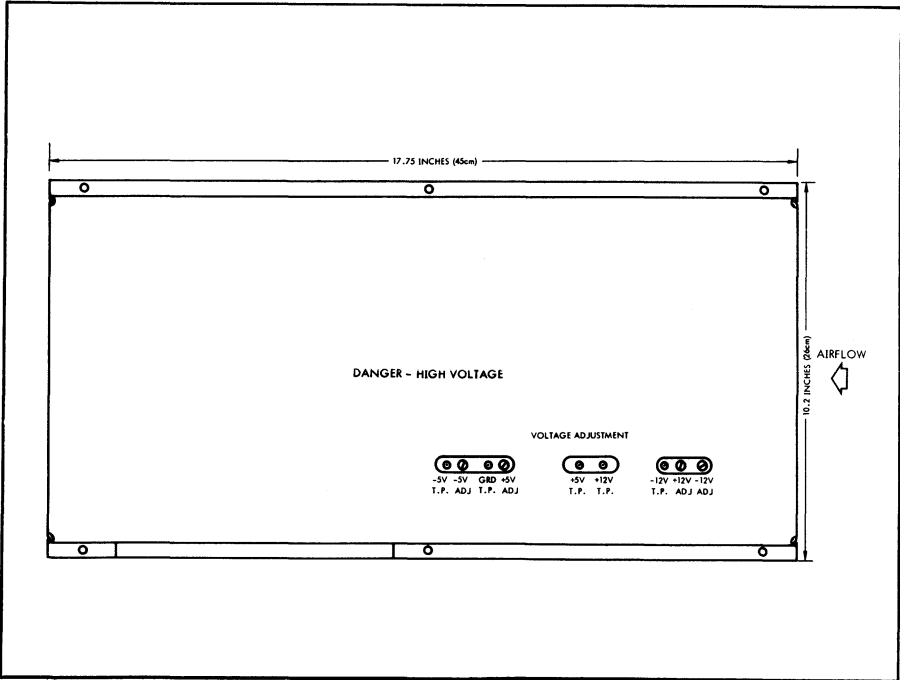
Additional peripheral controller cards require that the expansion chassis also include a left-half I/O backplane. This configuration provides 10 additional controller card slots, with slot 1 not wired, and an I/O termination shoe in slot 2.

Power Supply

The Varian 620/L-100 power supply provides power for the CPU, 32K of memory,

and all computer features and options. It is contained in a standard rack-mountable chassis, normally installed behind or near the mainframe or expansion chassis. A fan at one end of the power supply provides cooling for the electronic components.

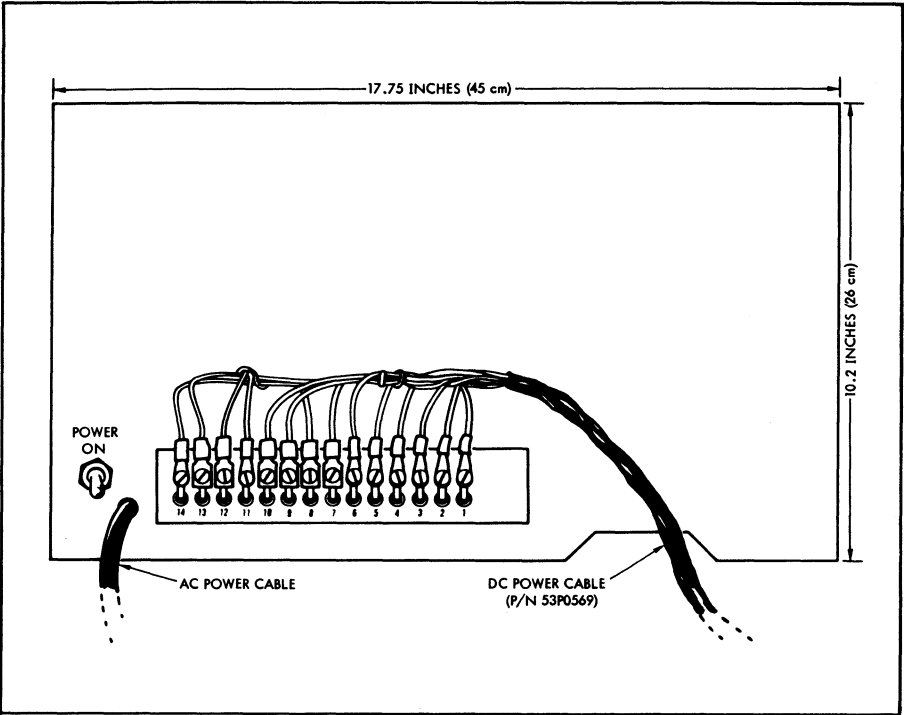
The top panel of the power supply (figure 33-8) has test points and adjustment controls for the +5, -5, +12, and -12 voltages. The bottom panel (figure 33-9) contains the POWER switch, the ac power cord, and the terminal board that connects to the dc power cable.



VTII-1154

Figure 33-8. Power Supply Top Panel

PHYSICAL CONFIGURATION



VTII-1155

Figure 33-9. Power Supply Bottom Panel

SECTION 34 - SYSTEM INTERCONNECION

The Varian 620/L-100 system is interconnected by cables that plug into connectors at the rear of the mainframe and expansion chassis. These connector panels distribute dc power and can be used for special I/O applications.

Connector Panels

Mainframe Connector Panel

The mainframe connector panel (figure 34-1) contains power connector J30 and Teletype connector J31.

An optional feature of the mainframe connector panel is an I/O harness assembly (part number 53P0571) wired between slot 26 and an optional I/O connector (J32). In this configuration, the user can connect existing equipment directly to the Varian 620/L-100 I/O lines (without costly modification). Additional connectors can

be added to the panel for special applications.

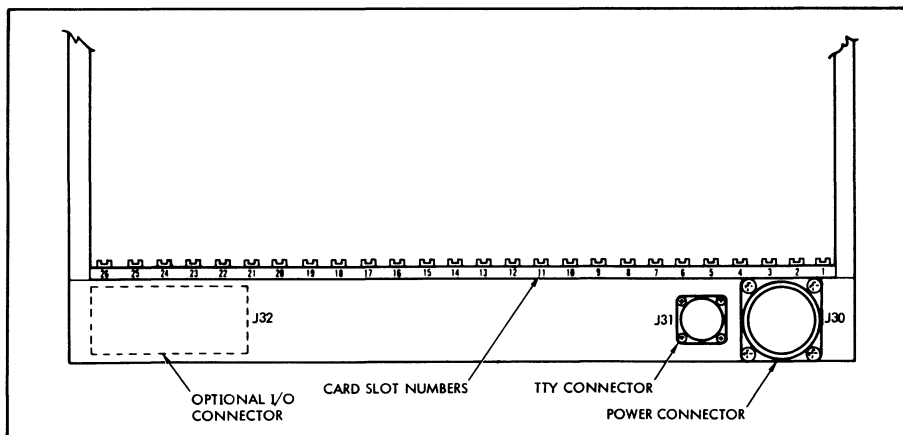
Expansion Chassis Connector Panel

The expansion chassis connector panel (figure 34-2) contains power connector J30, which routes power to expansion memories and peripheral controllers.

The optional I/O configuration described above can also be provided for expansion chassis.

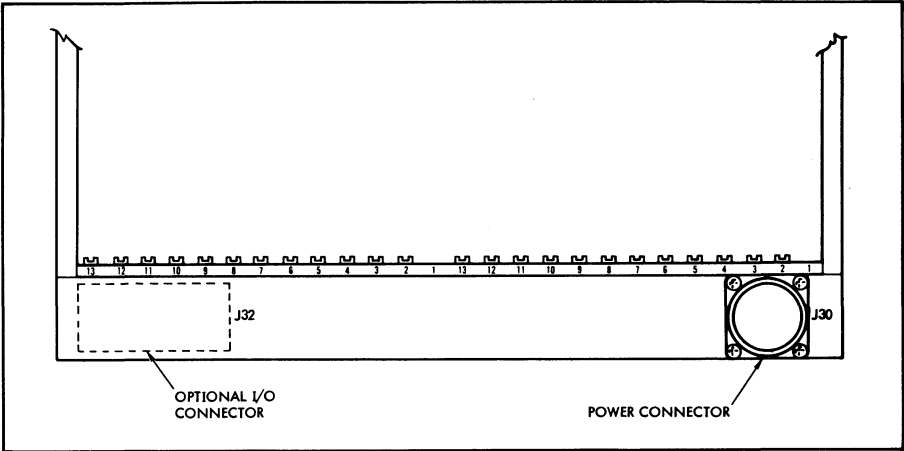
Power Supply Connector Panel

The power supply connector (pictured in section 33, figure 33-9) is located on the bottom panel of the chassis. It contains a three-wire ac power cable, a circuit-breaker switch, and a terminal board for interconnection with the dc power cable.



VTII-1152

Figure 34-1. Mainframe Connector Panel



VTII-1153

Figure 34-2. Expansion Chassis Connector Panel

The dc power cable (part number 53P0569) is 3 feet (1.8m) long and connects to J30 on the mainframe or expansion chassis connector panel. When one power supply accommodates both a mainframe and an expansion chassis, two cables connect to the terminal board.

Terminal board pin assignments for the power supply connector panel are given in table 34-1. Pin assignments for the J30 connector on the mainframe and expansion chassis panels are given in table 34-2.

Table 34-1. Power Supply Terminal Board Pin Assignments

Pin No.	Function	Pin No.	Function
1	115V fan	8	Common
2	115V fan	9	Common
3	AC return	10	- 12V
4	AC out	11	+ 5V
5	Relay return	12	+ 5V
6	Relay set	13	- 5V
7	+ 12V	14	Data guard

Table 34-2. J30 DC Power Cable Pin Assignments

Pin No.	Function	Pin No.	Function
1	----	10	Data guard
2	----	11	----
3	AC out	12	+ 12V
4	Relay return	13	- 12V
5	115V fan	14	+ 5V
6	115V fan	15	- 5V
7	Relay coil	16	Common
8	AC return	17	Common
9	----		

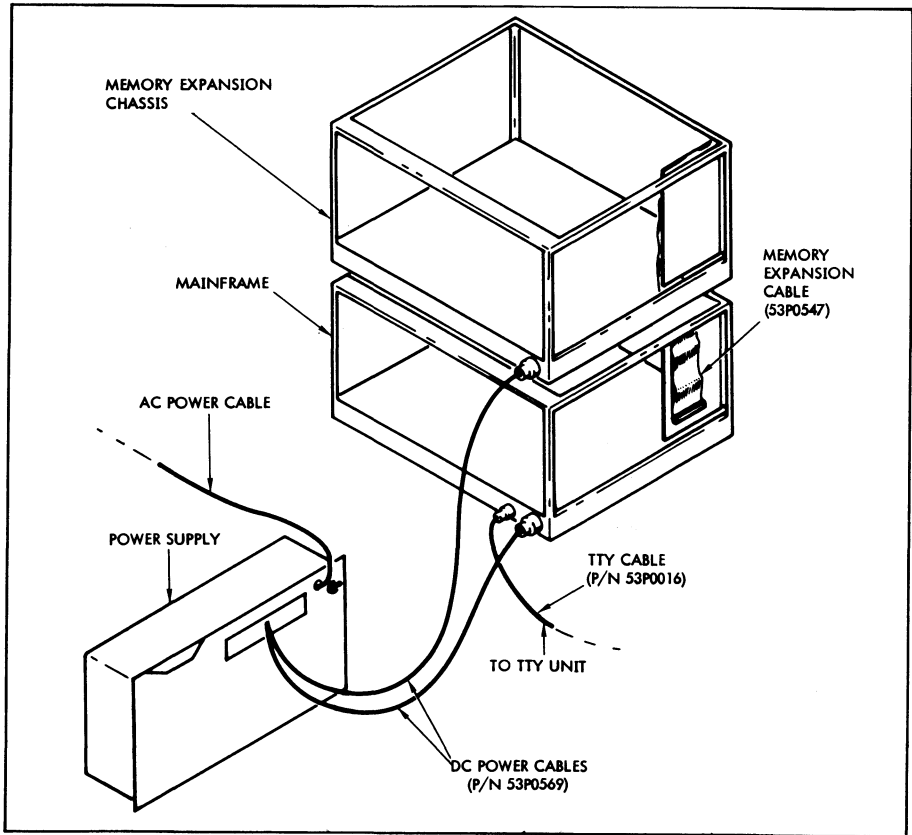


Figure 34-3. Memory Expansion Interconnection

Memory Expansion Interconnection

Memory expansion chassis are connected to the mainframe by a short, flat cable with 122-pin circuit-card connectors at both ends (figure 34-3). The cable (part number 53P0547) plugs into card slot 1 of the mainframe and card slot 2 of the expansion chassis.

Only one memory expansion chassis is required in even the fully expanded 32K system. The chassis must be located directly above or below the mainframe.

Memory expansion cable pin assignments are listed in table 34-3. Refer to chapter VIII of the Varian 620/L-100 Maintenance Manual (document number 98 A 9905 150) for descriptions of the signal mnemonics given in the table.

Table 34-3. Memory Expansion Cable Pin Assignments

Pin No.	Signal	Pin No.	Signal
1	GRD	33	GRD
2	----	34	----
3	GRD	35	GRD
4	L00X +	36	W00X -
5	GRD	37	GRD
6	L01X +	38	W01X -
7	GRD	39	GRD
8	L02X +	40	W02X -
9	GRD	41	GRD
10	L03X +	42	W03X -
11	GRD	43	GRD
12	L04X +	44	W04X -
13	GRD	45	GRD
14	L05X +	46	W05X -
15	GRD	47	GRD
16	L06X +	48	W06X -
17	GRD	49	GRD
18	L07X +	50	W07X -
19	GRD	51	GRD
20	L08X +	52	----
21	GRD	53	GRD
22	L09X +	54	----
23	GRD	55	GRD
24	L10X +	56	SASX -
25	GRD	57	GRD
26	L11X +	58	RXXx -
27	GRD	59	GRD
28	L12X +	60	WRTX +
29	GRD	61	GRD
30	L13X +	62	----
31	GRD	63	GRD
32	L14X +	64	SSL1 -

Table 34-3. Memory Expansion Cable Pin Assignments (continued)

Pin No.	Signal	Pin No.	Signal
65	GRD	94	RWTI -
66	SSL2 -	95	GRD
67	GRD	96	FCYX +
68	SSL3 -	97	GRD
69	GRD	98	TCRX -
70	SSL4 -	99	GRD
71	GRD	100	RWT2 -
72	W08X -	101	GRD
73	GRD	102	----
74	W09X -	103	GRD
75	GRD	104	RSTM -
76	W10X -	105	GRD
77	GRD	106	----
78	W11X -	107	GRD
79	GRD	108	----
80	W12X -	109	GRD
81	GRD	110	INHE - 1
82	W13X -	111	GRD
83	GRD	112	INHE - 0
84	W14X -	113	GRD
85	GRD	114	----
86	W15X -	115	GRD
87	GRD	116	----
88	MSCE +	117	GRD
89	GRD	118	----
90	MSPX +	119	GRD
91	GRD	120	----
92	WSTX -	121	GRD
93	GRD	122	GRD

I/O Expansion Interconnection

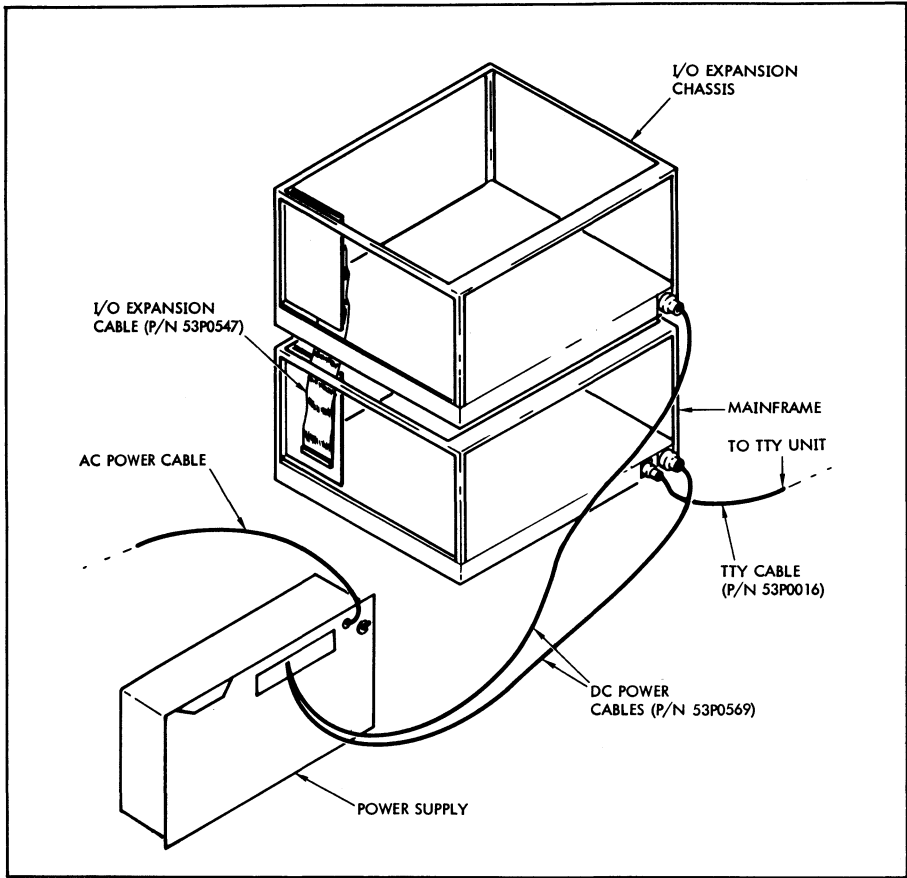
An I/O expansion chassis is connected to the mainframe by the same type of cable (part number 53P0547) used in memory expansion (figure 34-4). However, the I/O expansion cable is available in lengths up to 20 feet (6m).

The I/O expansion cable plugs into card slot 26 of the mainframe and slot 13 of the

right-hand I/O backplane in the expansion chassis.

An I/O termination shoe card (part number 44P0530) plugs into card slot 2 of the left-hand I/O backplane (if present).

Pin assignments for the I/O expansion cable are given in table 34-4. Chapter VII of the maintenance manual describes the signal mnemonics given in the table.



VTII-1157

Figure 34-4. I/O Expansion Interconnection

Table 34-4. I/O Expansion Cable Pin Assignments

Pin No.	Signal	Pin No.	Signal
1	GRD	6	EB02 - I
2	EB00 - I	7	RT03 -
3	RT01 -	8	EB03 -
4	EB01 - I	9	RT04 -
5	RT02 -	10	EB04 - I

Table 34-4. I/O Expansion Cable Pin Assignments (continued)

Pin No.	Signal	Pin No.	Signal
11	EB05 - I	40	RT12 -
12	EB06 - I	41	PR3X - I
13	EB07 - I	42	PR4X - I
14	EB08 - I	43	SYRT - I
15	EB09 - I	44	IUAX - I
16	EB10 - I	45	IUCX - I
17	EB11 - I	46	IURX - I
18	EB12 - I	47	IUJX - I
19	EB13 - I	48	GRD
20	EB14 - I	49	TRQX - B
21	EB15 - I	50	TROX - B
22	RT05 -	51	RT15 -
23	----	52	BCDX - B
24	RT06 -	53	RT16 -
25	----	54	CDCX - B
26	RT07 -	55	RT17 -
27	FRYX - I	56	DCEX - B
28	RT08 -	57	RT18 -
29	DRYX - I	58	TAKX - B
30	RT09 -	59	RT19 -
31	SERX - I	60	DESX - B
32	RT10 -	61-73	----
33	TPIX - I	74	DAGS
34	RT11 -	75-99	----
35	TPOX - I	100	GRD
36	RT12 -	101-115	----
37	PR1X - I	116	+3V dc
38	RT13 -	117-121	----
39	PR2X - I	122	GRD

Teletype Interconnection

The 20-foot (6m) Teletype cable (part number 53P0016) connects from J31 on the rear of the mainframe to the Teletype unit.

The cable connects to an S connector labeled 2 in the 33 ASR Teletype unit (right rear, top row, second connector from

the right). Pin assignments for the 33 ASR cable are given in table 34-5.

On the 35 ASR and 35 KSR Teletypes, the Teletype cable is wired directly to a power terminal board (TB) in the Teletype unit. The terminal board is located at the lower right rear of the cabinet behind the Teletype printing mechanism. Pin assignments for the 35 ASR and 35 KSR cables are given in table 34-6.

Table 34-5. Teletype Cable Pin Assignments

J31 End	Teletype End (P2)	Controller Pin	Signal
1	9	113	Return
2	6	112	Receive
3	8	114	Send

System Interrupt Interconnection

System interrupt priority is established by connecting the selected standard features and I/O options (sections 30 and 8) in a priority chain. The following can be included in the chain: PF/R, MP, RTC, PIM, and BIC.

Interrupt priority assignments are normally wired at the factory before the equipment is delivered. The following description is presented for the user who wishes to augment or change a system. Specific information concerning the unique qualities of each system is included with the installation instructions delivered with the equipment.

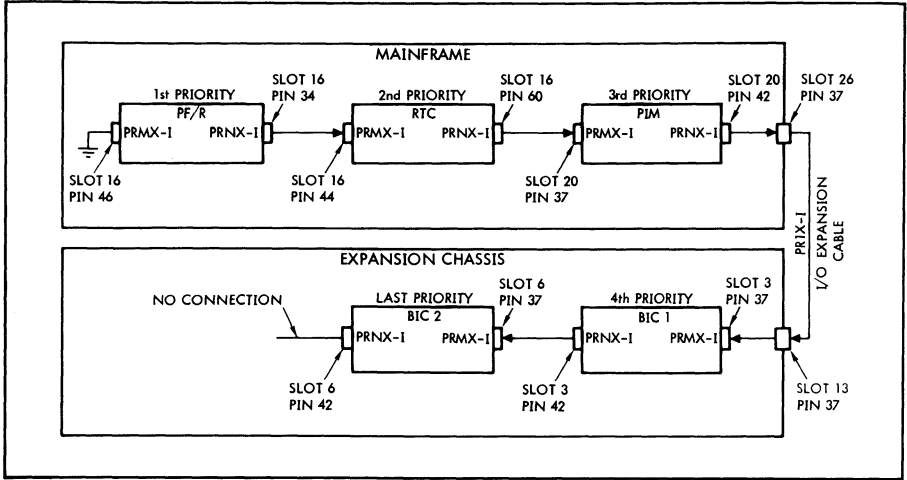
When all the options in the priority chain are in the same chassis, the priority-out signal (PRNX - I) of the highest-priority option is connected to the priority-in signal

(PRMX - I) of the next-lower-priority option. This process is repeated until all subsequent priorities within the chassis are assigned. The priority-in signal of the highest-priority option is connected to ground.

To establish a priority chain for options in separate chassis, four priority lines in the I/O expansion cable are wired into the computer as required. The designation and cable pin assignments of these lines are:

Signal	I/O Cable Pin
PR1X - I	37
PR2X - I	39
PR3X - I	41
PR4X - I	42

Figure 34-5 illustrates typical interrupt priority connections for a computer system containing the PF/R, RTC, and PIM in the mainframe, and two BICs in an expansion chassis.



VTII-1158

Figure 34-5. Typical System Interrupt Priority Connection

PIM Interconnection

The PIM connects to the CPU through the mainframe backplane. All signals enter and leave the PIM through receiver or driver stages that provide buffering between internal circuits and external lines.

The PIM has eight interrupt lines (IL00 through IL07) for connecting with up to eight peripheral controllers. PIM interconnection consists of connecting these lines to the selected peripheral controller cards to provide them with interrupt capabilities.

For peripheral controllers in the same chassis with the PIM, the interrupt lines are connected at the backplane. Controllers in other chassis require an interrupt cable. Card-edge connectors J1 and J2 on the PIM are parallel-wired to connect to two interrupt cables. The length of the cable is either 10 feet (3m) or 20 feet (6m).

Since the PIM is the termination for the interrupt cable, no termination shoe is required.

Table 34-7 lists the pin assignments for the interrupt lines.

Table 34-7. Interrupt Line Pin Assignments

Line	P1	J1	J2
IL00 -	108	3	3
IL01 -	114	13	13
IL02 -	104	9	9
IL03 -	110	15	15
IL04 -	102	11	11
IL05 -	88	5	5
IL06 -	112	1	1
IL07 -	86	7	7

BIC Interconnection

The BIC interfaces with the CPU and its associated peripheral controllers entirely through the backplane connectors. When two or more BICs are installed in the same chassis, the BIC control lines are connected only to the controller (or controllers) under the supervision of the individ-

ual BIC. BICs cannot be connected to each other. When a BIC-supervised controller is in a separate chassis, the control lines are extended via the I/O bus.

Specific BIC interconnection details are given in the BIC manual (document number 98 A 9902 423).

SECTION 35 - SYSTEM OPERATION

Program Execution

The Varian 620/L-100 requires very little preparation before a program can be executed. Assuming that the system, including peripherals, is properly installed and connected to an ac power source, the following procedure is followed to make a cold start (i.e., when a new system is being initialized or the contents of memory are unknown).

- a. Turn on computer power by placing the power keyswitch on the control panel (figure 35-1) to PWR ON.
- b. Initialize the system by pressing SYSTEM RESET, then reset all registers using the REGISTER selection and BIT RESET switches.
- c. Load the appropriate bootstrap loader routine (table 35-1) from the control panel.
- d. Load the binary load/dump program (BLD II, section 18) using the Teletype or high-speed paper tape reader (depending on the bootstrap loader selected). Verify after loading that the P register contains the proper starting address (section 18).
- e. Load the object program using the same paper tape reader as that used for BLD II.
- f. Press the RUN switch on the control panel.

This section describes control panel switches and indicators, and implementation of the above procedure and manual operations.

Switches and Indicators

The control panel of the Varian 620/L-100 is illustrated in figure 35-1.

Power Switch

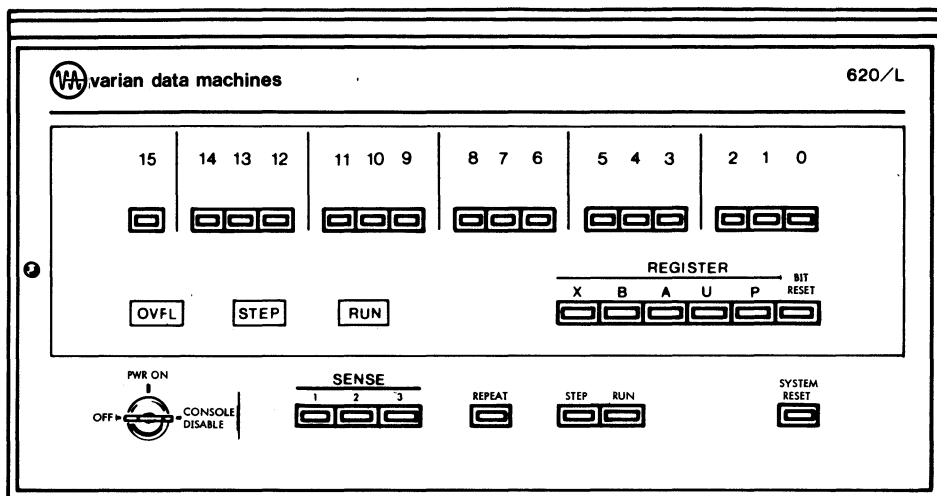
The key-operated power switch controls the ac input to the power supply.

In the OFF position, ac input to the power supply primary is disabled. In the PWR ON position, there is ac power to the power supply primary and the system should be fully operational. In the CONSOLE DISABLE position, there is ac power to the power supply primary and the computer is operational. However, all control panel switches are disabled except the power switch itself. Pressing any other switch while the power switch is in CONSOLE DISABLE has no effect. The control panel indicators are functional when the power switch is in either the PWR ON or CONSOLE DISABLE position.

The power switch key can be removed in any of the three positions. To turn off the computer, place the switch in PWR ON, press the STEP switch, then turn the power switch to OFF.

STEP Switch and Indicator

Pressing the momentary, spring-loaded STEP switch when the computer is in run mode (RUN indicator on) halts the computer after execution of the current instruction. Pressing STEP when the computer is halted executes the instruction currently in the instruction (U) register (step mode).



VT11-1162

Figure 35-1. Varian 620/L-100 Control Panel

When STEP is pressed, the STEP indicator lights; it goes out when the RUN switch is pressed.

RUN Switch and Indicator

Pressing the momentary, spring-loaded RUN switch executes the instruction currently in the U register and starts automatic processing of the stored program at the address specified by the P register.

When RUN is pressed, the RUN indicator lights; it goes out when the STEP switch is pressed.

REGISTER Selection Switches

Pressing one of the five toggle-action REGISTER selection switches selects the designated registers (X, B, A, U, and P) for display or entry.

Only one register can be selected at a time. Simultaneously pressing two or more

REGISTER switches disables the selection logic and the register display indicators.

Register Entry Switches and Display Indicators

The 16 indicators across the top of the control panel display the contents of a selected register. Data are entered into registers on the corresponding register entry switches located under the indicators. The indicators and switches are read from left to right (bits 15 through 0). A lighted indicator shows that that bit contains a one. For negative data, the sign bit (bit 15) is a one. The indicators and switches are divided into groups of three for ease in reading octal configurations.

To display the contents of a register, press STEP and select the desired register. The display indicators light when they correspond to register bits that contain ones. To remove the display, pull up on the REGISTER switch.

To enter data or instructions in a register:

- a. Display the contents of the selected register.
- b. Clear the register to all zeros by pressing the BIT RESET switch.
- c. Enter ones by pressing down on the register entry switches corresponding to the bits to be set. The associated display indicator lights for each switch pressed.

To enter data into computer memory:

- a. Load a storage instruction (e.g., STA, section 16) into the U register.
- b. Select the register specified by the storage instruction.
- c. Load the data word into the selected register using the data entry switches.
- d. Press STEP to execute the instruction in the U register. This stores the contents of the specified register at the effective memory address.

BIT RESET Switch

Pressing the momentary, spring-loaded BIT RESET switch when the computer is in step mode resets all bits of the selected register to zero. All register display indicators go out.

REPEAT Switch

The toggle-action REPEAT switch permits manual repetition of an instruction in the U register. When REPEAT is down, pressing STEP executes the instruction and advances the P register to the next program address. The U register contents

remain unchanged. REPEAT is disabled when the computer is in run mode.

SENSE Switches

The three toggle-action SENSE switches permit program modification by the operator. When the program contains instructions dependent on the setting of these switches, jumps and executions occur when the switch condition is met and do not occur when the condition is not met.

To set a SENSE switch, press down. To reset it, lift. Operations dependent on the position of this switch are executed if the switch is in the position indicated by the instruction.

EXAMPLE

A program can be written so that the operator can obtain a partial total of a column of figures being added by use of the JSS1 (jump if SENSE switch 1 is set, section 16) instruction. The program writes individual entries as long as SENSE switch 1 is not set. When the operator wants a partial total, he sets the switch. The program then jumps to an instruction sequence that prints the desired information.

SYSTEM RESET Switch

The momentary, spring-loaded SYSTEM RESET switch is used for initialization control and for stopping I/O operations. Pressing this switch halts the computer and initializes it and all peripherals. Note that *SYSTEM RESET* does not reset the registers.

SYSTEM OPERATION

OVFL (Overflow) Indicator

OVFL lights when a program overflow condition exists (section 16).

Manual Operations

Loading the Bootstrap Loader

After computer power is turned on and the

system initialized, load the bootstrap loader routine (table 35-1):

- In step mode, load a store A register relative to P instruction (054000) into the U register.
- Press the REPEAT switch.
- Load the starting memory address of the bootstrap loader (007756) into the P register.

Table 35-1. Bootstrap Loader Routines

Address	High-Speed Reader Code	Teletype Reader Code	Symbolic Coding		
007756	102637	102601	READ	CIB	RDR
007757	004011	004011		ASLB	NBIT - 7
007760	004041	004041		LRLB	1
007761	004446	004446		LLRL	6
007762	001020	001020		JBZ	SEL
007763	007772	007772		(Memory address)	
007764	055000	055000		STA	0,1
007765	001010	001010		JAZ	LHLT + 1
007766	007000*	007000		(Memory address)	
007767	005144	005144		IXR	
007770	005101	005101	ENTR	INCR	1
007771	100537	102601	SEL	SEL	RDON
007772	101537	101201		SEN	IBFR, READ
007773	007756	007756		(Memory address)	
007774	001000	001000		JMP	* - 2
007775	007772	007772		(Memory address)	

NOTE

The bootstrap loader routine is always loaded into the highest addresses of the first 4K memory increment, regardless of available memory. BLD II relocation and adaptation to the specified input device are described in section 18.

* Replace this code with 007600 if the test executive of MAINTAIN II (refer to document number 98 A 9952 060) is to be loaded and executed.

- d. Load the first bootstrap loader instruction into the A register. If the high-speed paper tape reader is to be used for subsequent program input, select the column headed High-Speed Reader Code in table 35-1; if using the Teletype paper tape reader, select the column headed Teletype Reader Code.
- e. Press STEP to load the A register contents into the address specified by the P register, which is incremented by one after the instruction is loaded.
- f. Clear the A register by pressing BIT RESET.
- g. Repeat steps d, e, and f for each bootstrap loader instruction.

To determine that the bootstrap loader is correctly loaded:

- a. Initialize the system by pressing SYSTEM RESET.
- b. Clear all registers by momentarily pressing each REGISTER selection switch, pressing BIT RESET each time.
- c. Load LDA instruction 014000 (load A register relative to P) into the U register.
- d. Load the bootstrap loader's starting memory address (007756) in the P register, keeping the REPEAT switch in the down position.
- e. Select the A register and press STEP. The contents of each memory address are displayed sequentially each time STEP is pressed.
- f. If an error is found, load the correct instruction code into memory.

NOTE

The P register error address is always the error address plus one.

The Varian 620/L-100 is now ready for use. The procedures for loading the binary load/dump program and subsequent object programs are given in section 18.

Loading, Displaying, and Altering Memory

To load data or instructions into memory, to display the contents of memory, or to alter the contents of memory, follow the procedures given in the description of the register entry switches and display indicators above.

Loading Sequential Memory Addresses

To load a sequential group of memory addresses, follow the procedures for loading the bootstrap loader routine using the A, B, or X register and loading the base address of the instruction group into the P register.

Displaying Sequential Memory Addresses

To display the contents of a group of sequential memory addresses:

- a. Press STEP and REPEAT.
- b. Load the base address of the instruction group into the P register.
- c. Load into the U register a relative-addressing load instruction (LDA, LDB, or LDX).
- d. Select the register specified by the instruction in step c.

SYSTEM OPERATION

- e. Press STEP once for each memory address to be displayed.

- e. Repeat step d once for each instruction.

Executing a Stored Program

To execute a stored program manually:

- a. In step mode, load the first address of the program into the P register.
- b. Clear the U register.
- c. Press STEP.
- d. Press STEP again to execute the instruction and to load the next instruction into the U register.

Repeating an Instruction

To repeat an instruction manually:

- a. In step mode, press REPEAT.
- b. Press STEP.

This procedure advances the P register each time STEP is pressed, but inhibits the loading of the U register with the next instruction in sequence.

APPENDIX A - INDEX OF INSTRUCTIONS

In the following, instruction mnemonics recognized only by the 620/f-100 computer are given in bold type.

Mnemonic	Octal Code	Description	Page
ADD	12xxxx	Add memory to A register	16-18
ADDE	00612x	Add extended	16-19
ADDI	006120	Add immediate	16-19
ANA	15xxxx	AND memory and A register	16-23
ANAE	00615x	AND extended	16-24
ANAI	006150	AND immediate	16-24
AOFA	005511	Add overflow to A register	16-14
AOFB	005522	Add overflow to B register	16-14
AOFX	005544	Add overflow to X register	16-14
ASLA	0042xx	Arithmetic shift left A register	16-9
ASLB	0040xx	Arithmetic shift left B register	16-9
ASRA	0043xx	Arithmetic shift right A register	16-8
ASRB	0041xx	Arithmetic shift right B register	16-8
BT	0064xx	Bit test	16-32
CIA	1025xx	Clear and input to A register	16-45
CIAB	1027xx	Clear and input to A and B registers	16-46
CIB	1026xx	Clear and input to B register	16-45
COMPL	005xxx	Complement source to destination registers	16-17
CPA	005211	Complement A register	16-14
CPB	005222	Complement B register	16-14
CPX	005244	Complement X register	16-14
DAR	005311	Decrement A register	16-13
DBR	005322	Decrement B register	16-13
DECR	005xxx	Decrement source to destination registers	16-17
DIV	17xxxx	Divide	16-20
DIVE	00617x	Divide extended	16-21

INDEX OF INSTRUCTIONS (continued)

Mnemonic	Octal Code	Description	Page
DIVI	006170	Divide immediate	16-21
DXR	005344	Decrement X register	16-14
ERA	13xxxx	Exclusive-OR memory and A register	16-22
ERAE	00613x	Exclusive-OR extended	16-23
ERAI	006130	Exclusive-OR immediate	16-23
EXC	100xxx	External control	16-45
HLT	000000	Halt	16-44
IAR	005111	Increment A register	16-13
IBR	005122	Increment B register	16-13
IJMP	0067xx	Indexed jump	16-26
IME	1020xx	Input to memory	16-46
INA	1021xx	Input to A register	16-46
INAB	1023xx	Input to A and B registers	16-46
INB	1022xx	Input to B register	16-46
INCR	005xxx	Increment source to destination registers	16-16
INR	04xxxx	Increment memory and replace	16-18
INRE	00604x	Increment memory and replace extended	16-18
INRI	006040	Increment memory and replace immediate	16-18
IXR	005144	Increment X register	16-13
JAN	001004	Jump if A register negative	16-27
JANM	002004	Jump and mark if A register negative	16-34
JANZ	001016	Jump if A register not zero	16-28
JANZM	002016	Jump and mark if A register not zero	16-35
JAP	001002	Jump if A register positive	16-27
JAPM	002002	Jump and mark if A register positive	16-34
JAZ	001010	Jump if A register zero	16-28
JAZM	002010	Jump and mark if A register zero	16-34
JBNZ	001026	Jump if B register not zero	16-28
JBNZM	002026	Jump and mark if B register not zero	16-35
JBZ	001020	Jump if B register zero	16-28

INDEX OF INSTRUCTIONS (continued)

Mnemonic	Octal Code	Description	Page
JBZM	002020	Jump and mark if B register zero	16-34
JIF	001xxx	Jump if conditions met	16-30
JIFM	002xxx	Jump and mark if conditions met	16-38
JMP	001000	Jump unconditionally	16-26
JMPM	002000	Jump and mark unconditionally	16-33
JO	001001	Jump if overflow indicator set	16-27
JOFN	001007	Jump if overflow indicator not set	16-27
JOFM	002001	Jump and mark if overflow indicator set	16-33
JOFNM	002007	Jump and mark if overflow indicator not set	16-33
JSR	0065xx	Jump unconditionally and set return in X register	16-31
JS1M	002100	Jump and mark if SENSE switch 1 set	16-36
JS2M	002200	Jump and mark if SENSE switch 2 set	16-36
JS3M	002400	Jump and mark if SENSE switch 3 set	16-37
JS1N	001106	Jump if SENSE switch 1 not set	16-30
JS2N	001206	Jump if SENSE switch 2 not set	16-30
JS3N	001406	Jump if SENSE switch 3 not set	16-30
JS1NM	002106	Jump and mark if SENSE switch 1 not set	16-37
JS2NM	002206	Jump and mark if SENSE switch 2 not set	16-37
JS3NM	002406	Jump and mark if SENSE switch 3 not set	16-38
JSS1	001100	Jump if SENSE switch 1 set	16-29
JSS2	001200	Jump if SENSE switch 2 set	16-29
JSS3	001400	Jump if SENSE switch 3 set	16-29
JXNZ	001046	Jump if X register not zero	16-29
JXNZM	002046	Jump and mark if X register not zero	16-36
JXZ	001040	Jump if X register zero	16-28
JXZM	002040	Jump and mark if X register zero	16-35
LASL	0044xx	Long arithmetic shift left	16-9

INDEX OF INSTRUCTIONS (continued)

Mnemonic	Octal Code	Description	Page
LASR	0045xx	Long arithmetic shift right	16-9
LDA	01xxxx	Load A register	16-2
LDAE	00601x	Load A register extended	16-3
LDAI	006010	Load A register immediate	16-5
LDB	02xxxx	Load B register	16-2
LDBE	00601x	Load B register extended	16-3
LDBI	006010	Load B register immediate	16-5
LDX	03xxxx	Load X register	16-2
LDXE	00603x	Load X register extended	16-4
LDXI	006030	Load X register immediate	16-5
LLRL	0044xx	Long logical rotation left	16-8
LLSR	0045xx	Long logical rotation right	16-8
LRLA	0042xx	Logical rotation left A register	16-7
LRLB	0040xx	Logical rotation left B register	16-7
LSRA	0043xx	Logical shift right A register	16-7
LSRB	0041xx	Logical shift right B register	16-7
MERG	005xxx	Merge source to destination registers	16-16
MUL	16xxxx	Multiply	16-20
MULE	00616x	Multiply extended	16-20
MULI	006160	Multiply immediate	16-20
NOP	005000	No operation	16-44
OAB	1033xx	Output inclusive-OR of A and B registers	16-46
OAR	1031xx	Output from A register	16-46
OBR	01032xx	Output from B register	16-46
OME	1030xx	Output from memory	16-47
ORA	11xxxx	Inclusive-OR memory and A register	16-21
ORAE	00611x	Inclusive-OR extended	16-22
ORAI	006110	Inclusive-OR immediate	16-22
ROF	007400	Reset overflow indicator	16-44
SEN	1010xx	Program sense	16-45
SOF	007401	Set overflow indicator	16-44
SOFA	005711	Subtract overflow from A register	16-15

INDEX OF INSTRUCTIONS (continued)

Mnemonic	Octal Code	Description	Page
SOFB	005722	Subtract overflow from B register	16-15
SOFX	005744	Subtract overflow from X register	16-15
SRE	0066xx	Skip if register equal	16-24
STA	05xxxx	Store A register	16-2
STAE	00605x	Store A register extended	16-4
STAI	006050	Store A register immediate	16-5
STB	06xxxx	Store B register	16-3
STBE	00606x	Store B register extended	16-4
STBI	006060	Store B register immediate	16-6
STX	07xxxx	Store X register	16-3
STXE	00607x	Store X register extended	16-4
STXI	006070	Store X register immediate	16-6
SUB	14xxxx	Subtract memory from A register	16-19
SUBE	00614x	Subtract extended	16-19
SUBI	006140	Subtract immediate	16-20
TAB	005012	Transfer A register to B register	16-11
TAX	005014	Transfer A register to X register	16-11
TBA	005021	Transfer B register to A register	16-11
TBX	005024	Transfer B register to X register	16-11
TSA	007402	Transfer switches to A register	16-12
TXA	005041	Transfer X register to A register	16-11
TXB	005042	Transfer X register to B register	16-12
TZA	005001	Transfer zero to A register	16-12
TZB	005002	Transfer zero to B register	16-12
TZX	005004	Transfer zero to X register	16-12
XAN	003004	Execute if A register negative	16-40
XANZ	003016	Execute if A register not zero	16-41
XAP	003002	Execute if A register positive	16-40
XAZ	003010	Execute if A register zero	16-40

INDEX OF INSTRUCTIONS

INDEX OF INSTRUCTIONS (continued)

Mnemonic	Octal Code	Description	Page
XBNZ	003026	Execute if B register not zero	16-41
XBZ	003020	Execute if B register zero	16-40
XEC	003000	Execute unconditionally	16-39
XIF	003xxx	Execute if conditions met	16-43
XOF	003001	Execute if overflow indicator set	16-39
XOFN	003007	Execute if overflow indicator not set	16-39
XS1	003100	Execute if SENSE switch 1 set	16-42
XS2	003200	Execute if SENSE switch 2 set	16-42
XS3	003400	Execute if SENSE switch 3 set	16-42
XS1N	003106	Execute if SENSE switch 1 not set	16-42
XS2N	003206	Execute if SENSE switch 2 not set	16-43
XS3N	003406	Execute if SENSE switch 3 not set	16-43
XXNZ	003046	Execute if X register not zero	16-41
XXZ	003040	Execute if X register zero	16-41
ZERO	005xxx	Zero (clear) registers	16-17

APPENDIX B - NUMBER SYSTEMS

Digital computers use a binary number system based on a count (radix) of two. The binary number system has simpler rules than the familiar decimal (radix of 10) number system, making it ideal for computers. The electronic components that make up a digital computer are inherently binary. A relay is either opened or closed; magnetic materials (tape or core) are magnetized in one direction or another; a vacuum tube or transistor is either fully conducting or nonconducting; an electrical pulse can be transmitted at a given time or it cannot be transmitted.

Binary System

In the decimal system, we think in "tens". For example, the number 35 means: $10 + 10 + 10 + 5 = 35$. Or 35 can be written as: $3(10) + 5(1) = 35$. Or 35 can be written in positional notation as: $3(10^1) + 5(10^0) = 35$. In the pure binary system, we deal with powers of two rather than powers of 10. The positions of the digits do not have the meaning of units, tens, hundreds, thousands, etc.; instead, these positions signify units, twos, fours, eights, sixteens, etc. The sum of these binary positions gives the same decimal sum.

Decimal values

32 16 8 4 2 1

Binary positional notational weight

2^5 2^4 2^3 2^2 2^1 2^0

Remembering that in the binary system we have only two marks, 0 and 1, we then convert the decimal number 35 to a binary number; reading from right to left, we place a one in the first, second, and sixth positions and zeros in the other three positions.

$$1(2^5) + 0(2^4) + 0(2^3) \\ + 0(2^2) + 1(2^1) + 1(2^0)$$

The resulting binary number is 100011, which is binary for decimal 35 ($32 + 0 + 0 + 0 + 2 + 1 = 35$).

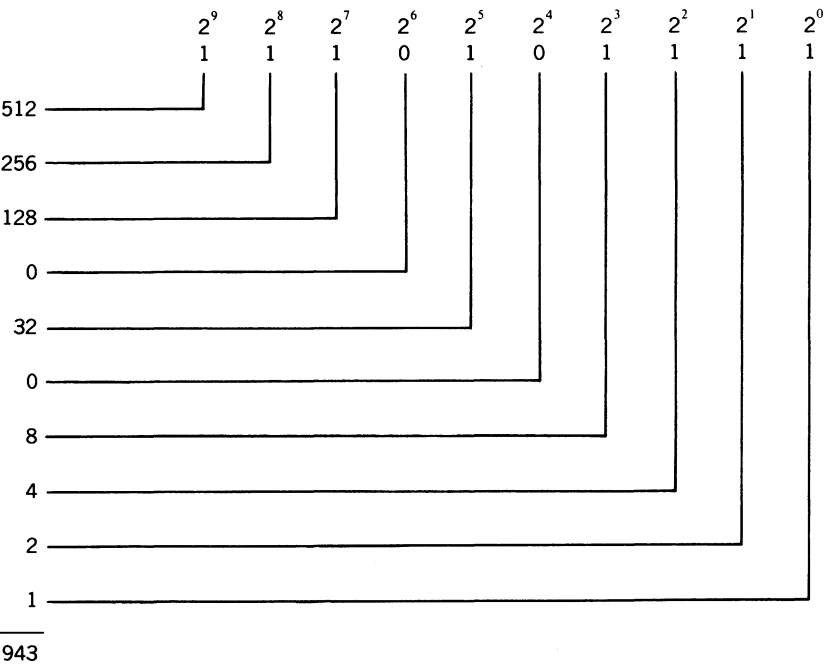
Decimal-to-Binary Conversion

Method 1. The positional notation chart used in the example above for converting decimal 35 to a binary number suggests a method for decimal-to-binary conversions. We ask the question, what is the largest number to the power of two that can be contained in the decimal number 35. The answer is 32, or 2^5 ; we place a one in that position. We next ask which power of two can be contained in the remainder $35 - 32$, or 3. Since 2^1 equals 2 and 2^0 equals 1 and both are contained in the remainder 3, we place ones in those positions. Hence, binary 100011 = decimal 35.

NUMBER SYSTEMS

Example

Convert decimal 943 to binary:

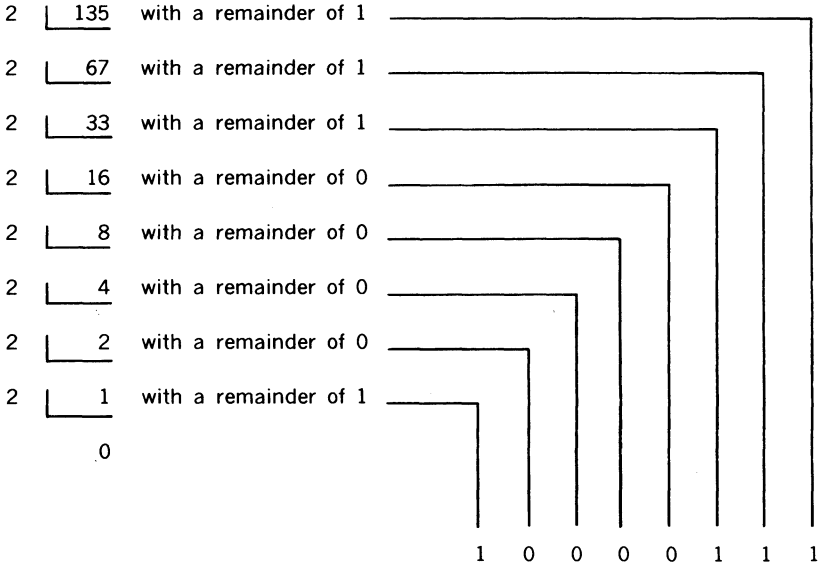


Method 2. A decimal number can be converted to its binary equivalent by successive division of the number by two. If there is a remainder after the first division, a binary one is placed in the least

significant (right-most) binary position. The occurrence or lack of a remainder after each division determines the binary state of each position.

Example

Convert decimal 135 to binary:



Binary-to-Decimal Conversion

Binary numbers are converted to their decimal equivalents by multiplying each

digit by two (starting with the most significant -- left-most -- digit) and adding the decimal value of the next digit to the right as illustrated below.

NUMBER SYSTEMS

Example

Convert binary 100001 to decimal:

1	0	0	0	0	1
x2					
2					
+0					
2					
x2					
4					
+0					
4					
x2					
8					
+0					
8					
x2					
16					
+0					
16					
x2					
32					
+1					
33					

Binary Addition

Only four rules apply in binary addition:

$$\begin{array}{ll} 0 + 0 = 0 & 1 + 0 = 1 \\ 0 + 1 = 1 & 1 + 1 = 10 \end{array}$$

Here 1 plus 1 is 10 (pronounced "one-oh") because binary 10 is decimal 2. This is the same as saying that decimal 1 plus 1 is 2. Following the above rules, it is possible to add any two binary numbers directly. For example, add decimal 12 and 5:

Decimal	Binary
12	01100
5	101
17	10001

To add 01011 and 00110:

Binary	Decimal
01011	11
00110	6
10001	17

In binary addition, there is the problem of the carry, as when $1 + 1 = 10$ -- that is, 0 plus carry 1. This is illustrated in the following example, where binary 111101 is added to 10110:

(A)	111101
(B)	10110
(C)	101011
(D)	11
(E)	1010011

The first step is to add A and B to get the partial sum C. Line D shows the two carries resulting from the $1 + 1$ sums. Adding partial sum C and the carries D produces the final sum E, or 1010011.

Binary Subtraction

Four rules apply in binary subtraction:

$$\begin{array}{rcl} 0 - 0 & = & 0 \\ 1 - 0 & = & 1 \\ 1 - 1 & = & 0 \\ 0 - 1 & = & 1 \end{array}$$

To subtract 1011 from 101101:

$$\begin{array}{r} 101101 \\ - 1011 \\ \hline 100010 \end{array}$$

Note that to subtract 1 from 0, it is necessary to borrow 1, making 1 from 10, or 1.

Complements also provide a means of subtraction. In the decimal system, the ten's complement is the difference between 10 and a given number -- hence, the complement of 7 is 3. The nine's complement is the difference between 9 and a given number, the complement of 7 being 2.

By adding complements, it is possible to subtract. To subtract, using the ten's complement system:

$$\begin{array}{r} 7 \\ +7 \quad (\text{ten's complement of } 3: \\ \hline 14 \end{array} \quad \begin{array}{l} 10 - 3 = 7 \end{array}$$

Delete the extra digit (which occurs because of the complement), giving the remainder 4 -- just as in the decimal system $7 - 3 = 4$.

Using the nine's complement system:

$$\begin{array}{r} 7 \\ +6 \quad (\text{nine's complement of } 3: \\ \hline 13 \end{array} \quad \begin{array}{l} 9 - 3 = 6 \end{array}$$

Here the extra 1 is not deleted, but is added to the 3, giving the same answer, 4. This adding of the extra digit, known as end-around carry, is a vital step in computer subtraction.

Since in the binary system there are only two digits, there can be only two complements. To find the one's complement of binary 1, subtract $1 - 1 = 0$. To find the one's complement of binary 0, subtract $1 - 0 = 1$. Thus, to find the complement of a binary number, change all ones to zeros and all zeros to ones; e.g., the complement of 1011 is 0100.

Thus, binary numbers can be subtracted directly:

$$\begin{array}{r} 1101 \\ - 1011 \\ \hline 0010 \end{array}$$

And, since the complement of 1011 is 0100, subtraction is also possible by adding complements:

$$\begin{array}{r} 1101 \\ + 0100 \\ \hline 10001 \\ \quad 1 \quad (\text{end-around carry}) \\ \hline 0010 \end{array}$$

NUMBER SYSTEMS

Binary Multiplication

Four rules apply in binary multiplication:

$$\begin{array}{ll} 0 \times 0 = 0 & 0 \times 1 = 0 \\ 1 \times 0 = 0 & 1 \times 1 = 1 \end{array}$$

No carries are considered in multiplication. Each digit of the multiplier is examined; when a one is found, the multiplicand is added to the result. When a zero is found in the multiplier, zeros are added to the result. The multiplicand is shifted left one digit for each multiplier digit.

Binary multiplication is thus a series of shifts and additions, as in the decimal system. For example, to multiply 100101 by 101:

$$\begin{array}{r} 100101 \\ 101 \\ \hline 100101 \\ 00000 \quad (\text{shift left, no add}) \\ 100101 \quad (\text{shift left and add}) \\ \hline 10111001 \quad (\text{sum}) \end{array}$$

For every 1 in the multiplier (101), the multiplicand (100101) is moved one place to the left and added. For every 0 in 101, there is one shift but no addition.

Binary Division

By applying the concepts of binary addition, subtraction, and multiplication, we can divide binary numbers. The divisor is subtracted from the dividend, and a 1 is placed in the quotient. If the divisor cannot be subtracted, a 0 is placed in the quotient.

To divide 101 into 1101010:

$$\begin{array}{r} 10101 \\ 101 \overline{) 1101010} \\ \underline{101} \\ 110 \\ \underline{101} \\ 110 \\ \underline{101} \\ 1 \end{array}$$

(remainder)

Binary-Coded Decimal (BCD) System

This system for representing decimal numbers expresses each decimal digit by a four-digit code called a word) written in binary notation:

BCD		Decimal
0000	=	0
0001	=	1
0010	=	2
0011	=	3
0100	=	4
0101	=	5
0110	=	6
0111	=	7
1000	=	8
1001	=	9
0001 0000	=	10

Thus, decimal 1971 would be expressed in BCD as:

0001	1001	0111	0001
1	9	7	1

Octal System

The octal system of assigning numerical values to binary forms is useful as a shorthand method of writing pure binary numbers. The octal system deals with groups of three binary positions; each group is considered a single digit. This means that, in any octal digit, there is a possibility of eight different binary configurations:

Binary		Octal
000	=	0
001	=	1
010	=	2
011	=	3
100	=	4
101	=	5
110	=	6
111	=	7

Given a series of binary digits, the first three to the left of the binary point are represented by the decimal notation 1, 2, 3..... 7×8^0 , the next three digits in order are represented decimally by 1, 2, 3..... 7×8^1 . As can be seen, each group of three binary bits represents some number (from 0 to 7) multiplied by a positional power of eight.

This binary number can be converted without using octal notation; however, the process requires the addition of seven quantities, instead of the three quantities in octal notation. To avoid confusion, octal numbers are designated with a leading zero, e.g., 0173.

Octal-to-Decimal Conversion

Octal representation can be converted to its decimal equivalent by multiplying each digit by eight (starting with the most significant -- left-most -- digit) and adding the decimal value of the next digit to the right as illustrated below.

Convert 0207 to decimal:

2	0	7
$\times 8$		
16		
+0		
128		
+7		
135		

Binary Groups	001	111	011				
Octal Notation	1	7	3				
Octal Equivalents	(1x8 ²)	(7x8 ¹)	(3x8 ⁰)				
Decimal Equivalents	64	+	56	+	3	=	123

NUMBER SYSTEMS

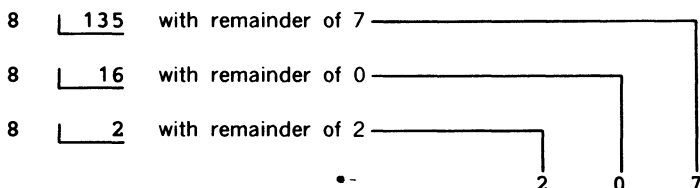
Decimal-to-Octal Conversion

A decimal number can be converted to its octal equivalent by dividing the decimal

number by eight and developing the octal number from the remainder as illustrated below.

Example

Convert decimal 135 to octal:



Hexadecimal System

Hexadecimal data are expressed to the radix (base) 16 and are related to the decimal numbers as follows:

Decimal	Hexadecimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000

Decimal	Hexadecimal	Binary
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Thus, hexadecimal numbers proceed from 0 through F (0 through 15 decimal), 10 through 1F (16 through 32 decimal), 20 through 2F (33 through 48 decimal), etc. To avoid confusion, hexadecimal numbers are designated with a leading dollar sign, e.g., \$0F3C.

Hexadecimal-to-Decimal Conversion

A hexadecimal number can be converted to its decimal equivalent by expanding each position. For example,

$$\begin{aligned} \$55F &= (5 \times 16^2) + (5 \times 16^1) + (15 \times 16^0) \\ &= (5 \times 256) + (5 \times 16) + (15 \times 1) \\ &= 1280 + 80 + 15 \\ &= 1375 \end{aligned}$$

APPENDIX C - POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125

APPENDIX D - OCTAL/DECIMAL INTEGER CONVERSIONS

0000
to
0777
(Octal)

0000
to
0511
(Decimal)

10000 - 4096

20000 - 8192

30000 - 12288

40000 - 16384

50000 - 20480

60000 - 24576

70000 - 28672

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0004	0005	0006	0007
0010	0008	0009	0010	0011	0012	0013	0014	0015
0020	0016	0017	0018	0019	0020	0021	0022	0023
0030	0024	0025	0026	0027	0028	0029	0030	0031
0040	0032	0033	0034	0035	0036	0037	0038	0039
0050	0040	0041	0042	0043	0044	0045	0046	0047
0060	0048	0049	0050	0051	0052	0053	0054	0055
0070	0056	0057	0058	0059	0060	0061	0062	0063
0100	0064	0065	0066	0067	0068	0069	0070	0071
0110	0072	0073	0074	0075	0076	0077	0078	0079
0120	0080	0081	0082	0083	0084	0085	0086	0087
0130	0088	0089	0090	0091	0092	0093	0094	0095
0140	0096	0097	0098	0099	0100	0101	0102	0103
0150	0104	0105	0106	0107	0108	0109	0110	0111
0160	0112	0113	0114	0115	0116	0117	0118	0119
0170	0120	0121	0122	0123	0124	0125	0126	0127
0200	0128	0129	0130	0131	0132	0133	0134	0135
0210	0136	0137	0138	0139	0140	0141	0142	0143
0220	0144	0145	0146	0147	0148	0149	0150	0151
0230	0152	0153	0154	0155	0156	0157	0158	0159
0240	0160	0161	0162	0163	0164	0165	0166	0167
0250	0168	0169	0170	0171	0172	0173	0174	0175
0260	0176	0177	0178	0179	0180	0181	0182	0183
0270	0184	0185	0186	0187	0188	0189	0190	0191
0300	0192	0193	0194	0195	0196	0197	0198	0199
0310	0200	0201	0202	0203	0204	0205	0206	0207
0320	0208	0209	0210	0211	0212	0213	0214	0215
0330	0216	0217	0218	0219	0220	0221	0222	0223
0340	0224	0225	0226	0227	0228	0229	0230	0231
0350	0232	0233	0234	0235	0236	0237	0238	0239
0360	0240	0241	0242	0243	0244	0245	0246	0247
0370	0248	0249	0250	0251	0252	0253	0254	0255
0400	0256	0257	0258	0259	0260	0261	0262	0263
0410	0264	0265	0266	0267	0268	0269	0270	0271
0420	0272	0273	0274	0275	0276	0277	0278	0279
0430	0280	0281	0282	0283	0284	0285	0286	0287
0440	0288	0289	0290	0291	0292	0293	0294	0295
0450	0296	0297	0298	0299	0300	0301	0302	0303
0460	0304	0305	0306	0307	0308	0309	0310	0311
0470	0312	0313	0314	0315	0316	0317	0318	0319
0500	0320	0321	0322	0323	0324	0325	0326	0327
0510	0328	0329	0330	0331	0332	0333	0334	0335
0520	0336	0337	0338	0339	0340	0341	0342	0343
0530	0344	0345	0346	0347	0348	0349	0350	0351
0540	0352	0353	0354	0355	0356	0357	0358	0359
0550	0360	0361	0362	0363	0364	0365	0366	0367
0560	0368	0369	0370	0371	0372	0373	0374	0375
0570	0376	0377	0378	0379	0380	0381	0382	0383
0600	0384	0385	0386	0387	0388	0389	0390	0391
0610	0392	0393	0394	0395	0396	0397	0398	0399
0620	0400	0401	0402	0403	0404	0405	0406	0407
0630	0408	0409	0410	0411	0412	0413	0414	0415
0640	0416	0417	0418	0419	0420	0421	0422	0423
0650	0424	0425	0426	0427	0428	0429	0430	0431
0660	0432	0433	0434	0435	0436	0437	0438	0439
0670	0440	0441	0442	0443	0444	0445	0446	0447
0700	0448	0449	0450	0451	0452	0453	0454	0455
0710	0456	0457	0458	0459	0460	0461	0462	0463
0720	0464	0465	0466	0467	0468	0469	0470	0471
0730	0472	0473	0474	0475	0476	0477	0478	0479
0740	0480	0481	0482	0483	0484	0485	0486	0487
0750	0488	0489	0490	0491	0492	0493	0494	0495
0760	0496	0497	0498	0499	0500	0501	0502	0503
0770	0504	0505	0506	0507	0508	0509	0510	0511

1000
to
1777
(Octal)

0512
to
1023
(Decimal)

	0	1	2	3	4	5	6	7
1000	0512	0513	0514	0515	0516	0517	0518	0519
1010	0520	0521	0522	0523	0524	0525	0526	0527
1020	0528	0529	0530	0531	0532	0533	0534	0535
1030	0536	0537	0538	0539	0540	0541	0542	0543
1040	0544	0545	0546	0547	0548	0549	0550	0551
1050	0552	0553	0554	0555	0556	0557	0558	0559
1060	0560	0561	0562	0563	0564	0565	0566	0567
1070	0568	0569	0570	0571	0572	0573	0574	0575
1100	0576	0577	0578	0579	0580	0581	0582	0583
1110	0584	0585	0586	0587	0588	0589	0590	0591
1120	0592	0593	0594	0595	0596	0597	0598	0599
1130	0600	0601	0602	0603	0604	0605	0606	0607
1140	0608	0609	0610	0611	0612	0613	0614	0615
1150	0616	0617	0618	0619	0620	0621	0622	0623
1160	0624	0625	0626	0627	0628	0629	0630	0631
1170	0632	0633	0634	0635	0636	0637	0638	0639
1200	0640	0641	0642	0643	0644	0645	0646	0647
1210	0648	0649	0650	0651	0652	0653	0654	0655
1220	0656	0657	0658	0659	0660	0661	0662	0663
1230	0664	0665	0666	0667	0668	0669	0670	0671
1240	0672	0673	0674	0675	0676	0677	0678	0679
1250	0680	0681	0682	0683	0684	0685	0686	0687
1260	0688	0689	0690	0691	0692	0693	0694	0695
1270	0696	0697	0698	0699	0700	0701	0702	0703
1300	0704	0705	0706	0707	0708	0709	0710	0711
1310	0712	0713	0714	0715	0716	0717	0718	0719
1320	0720	0721	0722	0723	0724	0725	0726	0727
1330	0728	0729	0730	0731	0732	0733	0734	0735
1340	0736	0737	0738	0739	0740	0741	0742	0743
1350	0744	0745	0746	0747	0748	0749	0750	0751
1360	0752	0753	0754	0755	0756	0757	0758	0759
1370	0760	0761	0762	0763	0764	0765	0766	0767
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865	0866	0867	0868	0869	0870	0871
1550	0872	0873	0874	0875	0876	0877	0878	0879
1560	0880	0881	0882	0883	0884	0885	0886	0887
1570	0888	0889	0890	0891	0892	0893	0894	0895
1600	0896	0897	0898	0899	0900	0901	0902	0903
1610	0904	0905	0906	0907	0908	0909	0910	0911
1620	0912	0913	0914	0915	0916	0917	0918	0919
1630	0920	0921	0922	0923	0924	0925	0926	0927
1640	0928	0929	0930	0931	0932	0933	0934	0935
1650	0936	0937	0938	0939	0940	0941	0942	0943
1660	0944	0945	0946	0947	0948	0949	0950	0951
1670	0952	0953	0954	0955	0956	0957	0958	0959
1700	0960	0961	0962	0963	0964	0965	0966	0967
1710	0968	0969	0970	0971	0972	0973	0974	0975
1720	0976	0977	0978	0979	0980	0981	0982	0983
1730	0984	0985	0986	0987	0988	0989	0990	0991
1740	0992	0993	0994	0995	0996	0997	0998	0999
1750	1000	1001	1002	1003	1004	1005	1006	1007
1760	1008	1009	1010	1011	1012	1013	1014	1015
1770	1016	1017	1018	1019	1020	1021	1022	1023

1000
to
1777
(Octal)

0512
to
1023
(Decimal)

	0	1	2	3	4	5	6	7
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865						

OCTAL/DECIMAL INTEGER CONVERSIONS

	0	1	2	3	4	5	6	7
2000	1024	1025	1026	1027	1028	1029	1030	1031
2010	1032	1033	1034	1035	1036	1037	1038	1039
2020	1040	1041	1042	1043	1044	1045	1046	1047
2030	1048	1049	1050	1051	1052	1053	1054	1055
2040	1056	1057	1058	1059	1060	1061	1062	1063
2050	1064	1065	1066	1067	1068	1069	1070	1071
2060	1072	1073	1074	1075	1076	1077	1078	1079
2070	1080	1081	1082	1083	1084	1085	1086	1087

2100	1088	1089	1090	1091	1092	1093	1094	1095
2110	1096	1097	1098	1099	1100	1101	1102	1103
2120	1104	1105	1106	1107	1108	1109	1110	1111
2130	1112	1113	1114	1115	1116	1117	1118	1119
2140	1120	1121	1122	1123	1124	1125	1126	1127
2150	1128	1129	1130	1131	1132	1133	1134	1135
2160	1136	1137	1138	1139	1140	1141	1142	1143
2170	1144	1145	1146	1147	1148	1149	1150	1151

2200	1152	1153	1154	1155	1156	1157	1158	1159
2210	1160	1161	1162	1163	1164	1165	1166	1167
2220	1168	1169	1170	1171	1172	1173	1174	1175
2230	1176	1177	1178	1179	1180	1181	1182	1183
2240	1184	1185	1186	1187	1188	1189	1190	1191
2250	1192	1193	1194	1195	1196	1197	1198	1199
2260	1200	1201	1202	1203	1204	1205	1206	1207
2270	1208	1209	1210	1211	1212	1213	1214	1215

2300	1216	1217	1218	1219	1220	1221	1222	1223
2310	1224	1225	1226	1227	1228	1229	1230	1231
2320	1232	1233	1234	1235	1236	1237	1238	1239
2330	1240	1241	1242	1243	1244	1245	1246	1247
2340	1248	1249	1250	1251	1252	1253	1254	1255
2350	1256	1257	1258	1259	1260	1261	1262	1263
2360	1264	1265	1266	1267	1268	1269	1270	1271
2370	1272	1273	1274	1275	1276	1277	1278	1279

	0	1	2	3	4	5	6	7
3000	1536	1537	1538	1539	1540	1541	1542	1543
3010	1544	1545	1546	1547	1548	1549	1550	1551
3020	1552	1553	1554	1555	1556	1557	1558	1559
3030	1560	1561	1562	1563	1564	1565	1566	1567
3040	1568	1569	1570	1571	1572	1573	1574	1575
3050	1576	1577	1578	1579	1580	1581	1582	1583
3060	1584	1585	1586	1587	1588	1589	1590	1591
3070	1592	1593	1594	1595	1596	1597	1598	1599

3100	1600	1601	1602	1603	1604	1605	1606	1607
3110	1608	1609	1610	1611	1612	1613	1614	1615
3120	1616	1617	1618	1619	1620	1621	1622	1623
3130	1624	1625	1626	1627	1628	1629	1630	1631
3140	1632	1633	1634	1635	1636	1637	1638	1639
3150	1640	1641	1642	1643	1644	1645	1646	1647
3160	1648	1649	1650	1651	1652	1653	1654	1655
3170	1656	1657	1658	1659	1660	1661	1662	1663

3200	1664	1665	1666	1667	1668	1669	1670	1671
3210	1672	1673	1674	1675	1676	1677	1678	1679
3220	1680	1681	1682	1683	1684	1685	1686	1687
3230	1688	1689	1690	1691	1692	1693	1694	1695
3240	1696	1697	1698	1699	1700	1701	1702	1703
3250	1704	1705	1706	1707	1708	1709	1710	1711
3260	1712	1713	1714	1715	1716	1717	1718	1719
3270	1720	1721	1722	1723	1724	1725	1726	1727

3300	1728	1729	1730	1731	1732	1733	1734	1735
3310	1736	1737	1738	1739	1740	1741	1742	1743
3320	1744	1745	1746	1747	1748	1749	1750	1751
3330	1752	1753	1754	1755	1756	1757	1758	1759
3340	1760	1761	1762	1763	1764	1765	1766	1767
3350	1768	1769	1770	1771	1772	1773	1774	1775
3360	1776	1777	1778	1779	1780	1781	1782	1783
3370	1784	1785	1786	1787	1788	1789	1790	1791

	0	1	2	3	4	5	6	7
2400	1280	1281	1282	1283	1284	1285	1286	1287
2410	1288	1289	1290	1291	1292	1293	1294	1295
2420	1296	1297	1298	1299	1300	1301	1302	1303
2430	1304	1305	1306	1307	1308	1309	1310	1311
2440	1312	1313	1314	1315	1316	1317	1318	1319
2450	1320	1321	1322	1323	1324	1325	1326	1327
2460	1328	1329	1330	1331	1332	1333	1334	1335
2470	1336	1337	1338	1339	1340	1341	1342	1343

2500	1344	1345	1346	1347	1348	1349	1350	1351
2510	1352	1353	1354	1355	1356	1357	1358	1359
2520	1360	1361	1362	1363	1364	1365	1366	1367
2530	1368	1369	1370	1371	1372	1373	1374	1375
2540	1376	1377	1378	1379	1380	1381	1382	1383
2550	1384	1385	1386	1387	1388	1389	1390	1391
2560	1392	1393	1394	1395	1396	1397	1398	1399
2570	1400	1401	1402	1403	1404	1405	1406	1407

2600	1408	1409	1410	1411	1412	1413	1414	1415
2610	1416	1417	1418	1419	1420	1421	1422	1423
2620	1424	1425	1426	1427	1428	1429	1430	1431
2630	1432	1433	1434	1435	1436	1437	1438	1439
2640	1440	1441	1442	1443	1444	1445	1446	1447
2650	1448	1449	1450	1451	1452	1453	1454	1455
2660	1456	1457	1458	1459	1460	1461	1462	1463
2670	1464	1465	1466	1467	1468	1469	1470	1471

2700	1472	1473	1474	1475	1476	1477	1478	1479
2710	1480	1481	1482	1483	1484	1485	1486	1487
2720	1488	1489	1490	1491	1492	1493	1494	1495
2730	1496	1497	1498	1499	1500	1501	1502	1503
2740	1504	1505	1506	1507	1508	1509	1510	1511
2750	1512	1513	1514	1515	1516	1517	1518	1519
2760	1520	1521	1522	1523	1524	1525	1526	1527
2770	1528	1529	1530	1531	1532	1533	1534	1535

	0	1	2	3	4	5	6	7
3400	1792	1793	1794	1795	1796	1797	1798	1799
3410	1800	1801	1802	1803	1804	1805	1806	1807
3420	1808	1809	1810	1811	1812	1813	1814	1815
3430	1816	1817	1818	1819	1820	1821	1822	1823
3440	1824	1825	1826	1827	1828	1829	1830	1831
3450	1832	1833	1834	1835	1836	1837	1838	1839
3460	1840	1841	1842	1843	1844	1845	1846	1847
3470	1848	1849	1850	1851	1852	1853	1854	1855

3500	1856	1857	1858	1859	1860	1861	1862	1863
3510	1864	1865	1866	1867	1868	1869	1870	1871
3520	1872	1873	1874	1875	1876	1877	1878	1879
3530	1880	1881	1882	1883	1884	1885	1886	1887
3540	1888	1889	1890	1891	1892	1893	1894	1895
3550	1896	1897	1898	1899	1900	1901	1902	1903
3560	1904	1905	1906	1907	1908	1909	1910	1911
3570	1912	1913	1914	1915	1916	1917	1918	1919

3600	1920	1921	1922	1923	1924	1925	1926	1927
3610	1928	1929	1930	1931	1932	1933	1934	1935
3620	1936	1937	1938	1939	1940	1941	1942	1943
3630	1944	1945	1946	1947	1948	1949	1950	1951
3640	1952	1953	1954	1955	1956	1957	1958	1959
3650	1960	1961	1962	1963	1964	1965	1966	1967
3660	1968	1969	1970	1971	1972	1973	1974	1975
3670	1976	1977	1978	1979	1980	1981	1982	1983

3700	1984	1985	1986	1987	1988	1989	1990	1991
3710	1992	1993	1994	1995	1996	1997	1998	1999
3720	2000	2001	2002	2003	2004	2005	2006	2007
3730	2008	2009	2010	2011	2012	2013	2014	2015
3740	2016	2017	2018	2019	2020	2021	2022	2023
3750	2024	2025	2026	2027	2028	2029	2030	2031
3760	2032	2033	2034	2035	2036	2037	2038	2039
3770	2040	2041	2042	2043	2044	2045	2046	2047

2000 to 2777 (Octal)
1024 to 1535 (Decimal)

Octal Decimal
10000 - 4096
20000 - 8192
30000 - 12288
40000 - 16384
50000 - 20480
60000 - 24576
70000 - 28672

3000 to 3777 (Octal)
1536 to 2047 (Decimal)

OCTAL/DECIMAL INTEGER CONVERSIONS

4000
to
4777
(Octal)

2048
to
2559
(Decimal)

Octal Decimal

10000 - 4096
20000 - 8192
30000 - 12288
40000 - 16384
50000 - 20480
60000 - 24576
70000 - 28672

	0	1	2	3	4	5	6	7
4000	2048	2049	2050	2051	2052	2053	2054	2055
4010	2056	2057	2058	2059	2060	2061	2062	2063
4020	2064	2065	2066	2067	2068	2069	2070	2071
4030	2072	2073	2074	2075	2076	2077	2078	2079
4040	2080	2081	2082	2083	2084	2085	2086	2087
4050	2088	2089	2090	2091	2092	2093	2094	2095
4060	2096	2097	2098	2099	2100	2101	2102	2103
4070	2104	2105	2106	2107	2108	2109	2110	2111
4100	2112	2113	2114	2115	2116	2117	2118	2119
4110	2120	2121	2122	2123	2124	2125	2126	2127
4120	2128	2129	2130	2131	2132	2133	2134	2135
4130	2136	2137	2138	2139	2140	2141	2142	2143
4140	2144	2145	2146	2147	2148	2149	2150	2151
4150	2152	2153	2154	2155	2156	2157	2158	2159
4160	2160	2161	2162	2163	2164	2165	2166	2167
4170	2168	2169	2170	2171	2172	2173	2174	2175
4200	2176	2177	2178	2179	2180	2181	2182	2183
4210	2184	2185	2186	2187	2188	2189	2190	2191
4220	2192	2193	2194	2195	2196	2197	2198	2199
4230	2200	2201	2202	2203	2204	2205	2206	2207
4240	2208	2209	2210	2211	2212	2213	2214	2215
4250	2216	2217	2218	2219	2220	2221	2222	2223
4260	2224	2225	2226	2227	2228	2229	2230	2231
4270	2232	2233	2234	2235	2236	2237	2238	2239
4300	2240	2241	2242	2243	2244	2245	2246	2247
4310	2248	2249	2250	2251	2252	2253	2254	2255
4320	2256	2257	2258	2259	2260	2261	2262	2263
4330	2264	2265	2266	2267	2268	2269	2270	2271
4340	2272	2273	2274	2275	2276	2277	2278	2279
4350	2280	2281	2282	2283	2284	2285	2286	2287
4360	2288	2289	2290	2291	2292	2293	2294	2295
4370	2296	2297	2298	2299	2300	2301	2302	2303

	0	1	2	3	4	5	6	7
4400	2304	2305	2306	2307	2308	2309	2310	2311
4410	2312	2313	2314	2315	2316	2317	2318	2319
4420	2320	2321	2322	2323	2324	2325	2326	2327
4430	2328	2329	2330	2331	2332	2333	2334	2335
4440	2336	2337	2338	2339	2340	2341	2342	2343
4450	2344	2345	2346	2347	2348	2349	2350	2351
4460	2352	2353	2354	2355	2356	2357	2358	2359
4470	2360	2361	2362	2363	2364	2365	2366	2367
4500	2368	2369	2370	2371	2372	2373	2374	2375
4510	2376	2377	2378	2379	2380	2381	2382	2383
4520	2384	2385	2386	2387	2388	2389	2390	2391
4530	2392	2393	2394	2395	2396	2397	2398	2399
4540	2400	2401	2402	2403	2404	2405	2406	2407
4550	2408	2409	2410	2411	2412	2413	2414	2415
4560	2416	2417	2418	2419	2420	2421	2422	2423
4570	2424	2425	2426	2427	2428	2429	2430	2431
4600	2432	2433	2434	2435	2436	2437	2438	2439
4610	2440	2441	2442	2443	2444	2445	2446	2447
4620	2448	2449	2450	2451	2452	2453	2454	2455
4630	2456	2457	2458	2459	2460	2461	2462	2463
4640	2464	2465	2466	2467	2468	2469	2470	2471
4650	2472	2473	2474	2475	2476	2477	2478	2479
4660	2480	2481	2482	2483	2484	2485	2486	2487
4670	2488	2489	2490	2491	2492	2493	2494	2495
4700	2496	2497	2498	2499	2500	2501	2502	2503
4710	2504	2505	2506	2507	2508	2509	2510	2511
4720	2512	2513	2514	2515	2516	2517	2518	2519
4730	2520	2521	2522	2523	2524	2525	2526	2527
4740	2528	2529	2530	2531	2532	2533	2534	2535
4750	2536	2537	2538	2539	2540	2541	2542	2543
4760	2544	2545	2546	2547	2548	2549	2550	2551
4770	2552	2553	2554	2555	2556	2557	2558	2559

5000
to
5777
(Octal)

2560
to
3071
(Decimal)

Octal Decimal

	0	1	2	3	4	5	6	7
5000	2560	2561	2562	2563	2564	2565	2566	2567
5010	2568	2569	2570	2571	2572	2573	2574	2575
5020	2576	2577	2578	2579	2580	2581	2582	2583
5030	2584	2585	2586	2587	2588	2589	2590	2591
5040	2592	2593	2594	2595	2596	2597	2598	2599
5050	2600	2601	2602	2603	2604	2605	2606	2607
5060	2608	2609	2610	2611	2612	2613	2614	2615
5070	2616	2617	2618	2619	2620	2621	2622	2623
5100	2624	2625	2626	2627	2628	2629	2630	2631
5110	2632	2633	2634	2635	2636	2637	2638	2639
5120	2640	2641	2642	2643	2644	2645	2646	2647
5130	2648	2649	2650	2651	2652	2653	2654	2655
5140	2656	2657	2658	2659	2660	2661	2662	2663
5150	2664	2665	2666	2667	2668	2669	2670	2671
5160	2672	2673	2674	2675	2676	2677	2678	2679
5170	2680	2681	2682	2683	2684	2685	2686	2687
5200	2688	2689	2690	2691	2692	2693	2694	2695
5210	2696	2697	2698	2699	2700	2701	2702	2703
5220	2704	2705	2706	2707	2708	2709	2710	2711
5230	2712	2713	2714	2715	2716	2717	2718	2719
5240	2720	2721	2722	2723	2724	2725	2726	2727
5250	2728	2729	2730	2731	2732	2733	2734	2735
5260	2736	2737	2738	2739	2740	2741	2742	2743
5270	2744	2745	2746	2747	2748	2749	2750	2751
5300	2752	2753	2754	2755	2756	2757	2758	2759
5310	2760	2761	2762	2763	2764	2765	2766	2767
5320	2768	2769	2770	2771	2772	2773	2774	2775
5330	2776	2777	2778	2779	2780	2781	2782	2783
5340	2784	2785	2786	2787	2788	2789	2790	2791
5350	2792	2793	2794	2795	2796	2797	2798	2799
5360	2800	2801	2802	2803	2804	2805	2806	2807
5370	2808	2809	2810	2811	2812	2813	2814	2815

	0	1	2	3	4	5	6	7
5400	2816	2817	2818	2819	2820	2821	2822	2823
5410	2824	2825	2826	2827	2828	2829	2830	2831
5420	2832	2833	2834	2835	2836	2837	2838	2839
5430	2840	2841	2842	2843	2844	2845	2846	2847
5440	2848	2849	2850	2851	2852	2853	2854	2855
5450	2856	2857	2858	2859	2860	2861	2862	2863
5460	2864	2865	2866	2867	2868	2869	2870	2871
5470	2872	2873	2874	2875	2876	2877	2878	2879
5500	2880	2881	2882	2883	2884	2885	2886	2887
5510	2888	2889	2890	2891	2892	2893	2894	2895
5520	2896	2897	2898	2899	2900	2901	2902	2903
5530	2904	2905	2906	2907	2908	2909	2910	2911
5540	2912	2913	2914	2915	2916	2917	2918	2919
5550	2920	2921	2922	2923	2924	2925	2926	2927
5560	2928	2929	2930	2931	2932	2933	2934	2935
5570	2936	2937	2938	2939	2940	2941	2942	2943
5600	2944	2945	2946	2947	2948	2949	2950	2951
5610	2952	2953	2954	2955	2956	2957	2958	2959
5620	2960	2961	2962	2963	2964	2965	2966	2967
5630	2968	2969	2970	2971	2972	2973	2974	2975
5640	2976	2977	2978	2979	2980	2981	2982	2983
5650	2984	2985	2986	2987	2988	2989	2990	2991
5660	2992	2993	2994	2995	2996	2997	2998	2999
5670	3000	3001	3002	3003	3004	3005	3006	3007
5700	3008	3009	3010	3011	3012	3013	3014	3015
5710	3016	3017	3018	3019	3020	3021	3022	3023
5720	3024	3025	3026	3027	3028	3029	3030	3031
5730	3032	3033	3034	3035	3036	3037	3038	3039
5740	3040	3041	3042	3043	3044	3045	3046	3047
5750	3048	3049	3050	3051	3052	3053	3054	3055
5760	3056	3057	3058	3059	3060	3061	3062	3063
5770	3064	3065	3066	3067	3068	3069	3070	3071

OCTAL/DECIMAL INTEGER CONVERSIONS

	0	1	2	3	4	5	6	7
6000	3072	3073	3074	3075	3076	3077	3078	3079
6010	3080	3081	3082	3083	3084	3085	3086	3087
6020	3088	3089	3090	3091	3092	3093	3094	3095
6030	3096	3097	3098	3099	3100	3101	3102	3103
6040	3104	3105	3106	3107	3108	3109	3110	3111
6050	3112	3113	3114	3115	3116	3117	3118	3119
6060	3120	3121	3122	3123	3124	3125	3126	3127
6070	3128	3129	3130	3131	3132	3133	3134	3135
6100	3136	3137	3138	3139	3140	3141	3142	3143
6110	3144	3145	3146	3147	3148	3149	3150	3151
6120	3152	3153	3154	3155	3156	3157	3158	3159
6130	3160	3161	3162	3163	3164	3165	3166	3167
6140	3168	3169	3170	3171	3172	3173	3174	3175
6150	3176	3177	3178	3179	3180	3181	3182	3183
6160	3184	3185	3186	3187	3188	3189	3190	3191
6170	3192	3193	3194	3195	3196	3197	3198	3199
6200	3200	3201	3202	3203	3204	3205	3206	3207
6210	3208	3209	3210	3211	3212	3213	3214	3215
6220	3216	3217	3218	3219	3220	3221	3222	3223
6230	3224	3225	3226	3227	3228	3229	3230	3231
6240	3232	3233	3234	3235	3236	3237	3238	3239
6250	3240	3241	3242	3243	3244	3245	3246	3247
6260	3248	3249	3250	3251	3252	3253	3254	3255
6270	3256	3257	3258	3259	3260	3261	3262	3263
6300	3264	3265	3266	3267	3268	3269	3270	3271
6310	3272	3273	3274	3275	3276	3277	3278	3279
6320	3280	3281	3282	3283	3284	3285	3286	3287
6330	3288	3289	3290	3291	3292	3293	3294	3295
6340	3296	3297	3298	3299	3300	3301	3302	3303
6350	3304	3305	3306	3307	3308	3309	3310	3311
6360	3312	3313	3314	3315	3316	3317	3318	3319
6370	3320	3321	3322	3323	3324	3325	3326	3327

	0	1	2	3	4	5	6	7
7000	3584	3585	3586	3587	3588	3589	3590	3591
7010	3592	3593	3594	3595	3596	3597	3598	3599
7020	3600	3601	3602	3603	3604	3605	3606	3607
7030	3608	3609	3610	3611	3612	3613	3614	3615
7040	3616	3617	3618	3619	3620	3621	3622	3623
7050	3624	3625	3626	3627	3628	3629	3630	3631
7060	3632	3633	3634	3635	3636	3637	3638	3639
7070	3640	3641	3642	3643	3644	3645	3646	3647
7100	3648	3649	3650	3651	3652	3653	3654	3655
7110	3656	3657	3658	3659	3660	3661	3662	3663
7120	3664	3665	3666	3667	3668	3669	3670	3671
7130	3672	3673	3674	3675	3676	3677	3678	3679
7140	3680	3681	3682	3683	3684	3685	3686	3687
7150	3688	3689	3690	3691	3692	3693	3694	3695
7160	3696	3697	3698	3699	3700	3701	3702	3703
7170	3704	3705	3706	3707	3708	3709	3710	3711
7200	3712	3713	3714	3715	3716	3717	3718	3719
7210	3720	3721	3722	3723	3724	3725	3726	3727
7220	3728	3729	3730	3731	3732	3733	3734	3735
7230	3736	3737	3738	3739	3740	3741	3742	3743
7240	3744	3745	3746	3747	3748	3749	3750	3751
7250	3752	3753	3754	3755	3756	3757	3758	3759
7260	3760	3761	3762	3763	3764	3765	3766	3767
7270	3768	3769	3770	3771	3772	3773	3774	3775
7300	3776	3777	3778	3779	3780	3781	3782	3783
7310	3784	3785	3786	3787	3788	3789	3790	3791
7320	3792	3793	3794	3795	3796	3797	3798	3799
7330	3800	3801	3802	3803	3804	3805	3806	3807
7340	3808	3809	3810	3811	3812	3813	3814	3815
7350	3816	3817	3818	3819	3820	3821	3822	3823
7360	3824	3825	3826	3827	3828	3829	3830	3831
7370	3832	3833	3834	3835	3836	3837	3838	3839

	0	1	2	3	4	5	6	7
6400	3328	3329	3330	3331	3332	3333	3334	3335
6410	3336	3337	3338	3339	3340	3341	3342	3343
6420	3344	3345	3346	3347	3348	3349	3350	3351
6430	3352	3353	3354	3355	3356	3357	3358	3359
6440	3360	3361	3362	3363	3364	3365	3366	3367
6450	3368	3369	3370	3371	3372	3373	3374	3375
6460	3376	3377	3378	3379	3380	3381	3382	3383
6470	3384	3385	3386	3387	3388	3389	3390	3391
6500	3392	3393	3394	3395	3396	3397	3398	3399
6510	3400	3401	3402	3403	3404	3405	3406	3407
6520	3408	3409	3410	3411	3412	3413	3414	3415
6530	3416	3417	3418	3419	3420	3421	3422	3423
6540	3424	3425	3426	3427	3428	3429	3430	3431
6550	3432	3433	3434	3435	3436	3437	3438	3439
6560	3440	3441	3442	3443	3444	3445	3446	3447
6570	3448	3449	3450	3451	3452	3453	3454	3455
6600	3456	3457	3458	3459	3460	3461	3462	3463
6610	3464	3465	3466	3467	3468	3469	3470	3471
6620	3472	3473	3474	3475	3476	3477	3478	3479
6630	3480	3481	3482	3483	3484	3485	3486	3487
6640	3488	3489	3490	3491	3492	3493	3494	3495
6650	3496	3497	3498	3499	3500	3501	3502	3503
6660	3504	3505	3506	3507	3508	3509	3510	3511
6670	3512	3513	3514	3515	3516	3517	3518	3519
6700	3520	3521	3522	3523	3524	3525	3526	3527
6710	3528	3529	3530	3531	3532	3533	3534	3535
6720	3536	3537	3538	3539	3540	3541	3542	3543
6730	3544	3545	3546	3547	3548	3549	3550	3551
6740	3552	3553	3554	3555	3556	3557	3558	3559
6750	3560	3561	3562	3563	3564	3565	3566	3567
6760	3568	3569	3570	3571	3572	3573	3574	3575
6770	3576	3577	3578	3579	3580	3581	3582	3583

	0	1	2	3	4	5	6	7
7400	3840	3841	3842	3843	3844	3845	3846	3847
7410	3848	3849	3850	3851	3852	3853	3854	3855
7420	3856	3857	3858	3859	3860	3861	3862	3863
7430	3864	3865	3866	3867	3868	3869	3870	3871
7440	3872	3873	3874	3875	3876	3877	3878	3879
7450	3880	3881	3882	3883	3884	3885	3886	3887
7460	3888	3889	3890	3891	3892	3893	3894	3895
7470	3896	3897	3898	3899	3900	3901	3902	3903
7500	3904	3905	3906	3907	3908	3909	3910	3911
7510	3912	3913	3914	3915	3916	3917	3918	3919
7520	3920	3921	3922	3923	3924	3925	3926	3927
7530	3928	3929	3930	3931	3932	3933	3934	3935
7540	3936	3937	3938	3939	3940	3941	3942	3943
7550	3944	3945	3946	3947	3948	3949	3950	3951
7560	3952	3953	3954	3955	3956	3957	3958	3959
7570	3960	3961	3962	3963	3964	3965	3966	3967
7600	3968	3969	3970	3971	3972	3973	3974	3975
7610	3976	3977	3978	3979	3980	3981	3982	3983
7620	3984	3985	3986	3987	3988	3989	3990	3991
7630	3992	3993	3994	3995	3996	3997	3998	3999
7640	4000	4001	4002	4003	4004	4005	4006	4007
7650	4008	4009	4010	4011	4012	4013	4014	4015
7660	4016	4017	4018	4019	4020	4021	4022	4023
7670	4024	4025	4026	4027	4028	4029	4030	4031
7700	4032	4033	4034	4035	4036	4037	4038	4039
7710	4040	4041	4042	4043	4044	4045	4046	4047
7720	4048	4049	4050	4051	4052	4053	4054	4055
7730	4056	4057	4058	4059	4060	4061	4062	4063
7740	4064	4065	4066	4067	4068	4069	4070	4071
7750	4072	4073	4074	4075	4076	4077	4078	4079
7760	4080	4081	4082	4083	4084	4085	4086	4087
7770	4088	4089	4090	4091	4092	4093	4094	4095

6000 to 6777
3072 to 3583
(Octal) (Decimal)

Octal Decimal
10000 - 4096
20000 - 8192
30000 - 12288
40000 - 16384
50000 - 20480
60000 - 24576
70000 - 28672

7000 to 7777
3584 to 4095
(Octal) (Decimal)

APPENDIX E - OCTAL/DECIMAL FRACTION CONVERSIONS

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000	.000000	.100	.125000	.200	.250000	.300	.375000
.001	.001953	.101	.126953	.201	.251953	.301	.376953
.002	.003906	.102	.128906	.202	.253906	.302	.378906
.003	.005859	.103	.130859	.203	.255859	.303	.380859
.004	.007812	.104	.132812	.204	.257812	.304	.382812
.005	.009765	.105	.134765	.205	.259765	.305	.384765
.006	.011718	.106	.136718	.206	.261718	.306	.386718
.007	.013671	.107	.138671	.207	.263671	.307	.388671
.010	.015625	.110	.140625	.210	.265625	.310	.390625
.011	.017578	.111	.142578	.211	.267578	.311	.392578
.012	.019531	.112	.144531	.212	.269531	.312	.394531
.013	.021484	.113	.146484	.213	.271484	.313	.396484
.014	.023437	.114	.148437	.214	.273437	.314	.398437
.015	.025390	.115	.150390	.215	.275390	.315	.400390
.016	.027343	.116	.152343	.216	.277343	.316	.402343
.017	.029296	.117	.154296	.217	.279296	.317	.404296
.020	.031250	.120	.156250	.220	.281250	.320	.406250
.021	.033203	.121	.158203	.221	.283203	.321	.408203
.022	.035156	.122	.160156	.222	.285156	.322	.410156
.023	.037109	.123	.162109	.223	.287109	.323	.412109
.024	.039062	.124	.164062	.224	.289062	.324	.414062
.025	.041015	.125	.166015	.225	.291015	.325	.416015
.026	.042968	.126	.167968	.226	.292968	.326	.417968
.027	.044921	.127	.169921	.227	.294921	.327	.419921
.030	.046875	.130	.171875	.230	.296875	.330	.421875
.031	.048828	.131	.173828	.231	.298828	.331	.423828
.032	.050781	.132	.175781	.232	.300781	.332	.425781
.033	.052734	.133	.177734	.233	.302734	.333	.427734
.034	.054687	.134	.179687	.234	.304687	.334	.429687
.035	.056640	.135	.181640	.235	.306640	.335	.431640
.036	.058593	.136	.183593	.236	.308593	.336	.433593
.037	.060546	.137	.185546	.237	.310546	.337	.435546
.040	.062500	.140	.187500	.240	.312500	.340	.437500
.041	.064453	.141	.189453	.241	.314453	.341	.439453
.042	.066406	.142	.191406	.242	.316406	.342	.441406
.043	.068359	.143	.193359	.243	.318359	.343	.443359
.044	.070312	.144	.195312	.244	.320312	.344	.445312
.045	.072265	.145	.197265	.245	.322265	.345	.447265
.046	.074218	.146	.199218	.246	.324218	.346	.449218
.047	.076171	.147	.201171	.247	.326171	.347	.451171
.050	.078125	.150	.203125	.250	.328125	.350	.453125
.051	.080078	.151	.205078	.251	.330078	.351	.455078
.052	.082031	.152	.207031	.252	.332031	.352	.457031
.053	.083984	.153	.208984	.253	.333984	.353	.458984
.054	.085937	.154	.210937	.254	.335937	.354	.460937
.055	.087890	.155	.212890	.255	.337890	.355	.462890
.056	.089843	.156	.214843	.256	.339843	.356	.464843
.057	.091796	.157	.216796	.257	.341796	.357	.466796
.060	.093750	.160	.218750	.260	.343750	.360	.468750
.061	.095703	.161	.220703	.261	.345703	.361	.470703
.062	.097656	.162	.222656	.262	.347656	.362	.472656
.063	.099609	.163	.224609	.263	.349609	.363	.474609
.064	.101562	.164	.226562	.264	.351562	.364	.476562
.065	.103515	.165	.228515	.265	.353515	.365	.478515
.066	.105468	.166	.230468	.266	.355468	.366	.480468
.067	.107421	.167	.232421	.267	.357421	.367	.482421
.070	.109375	.170	.234375	.270	.359375	.370	.484375
.071	.111328	.171	.236328	.271	.361328	.371	.486328
.072	.113281	.172	.238281	.272	.363281	.372	.488281
.073	.115234	.173	.240234	.273	.365234	.373	.490234
.074	.117187	.174	.242187	.274	.367187	.374	.492187
.075	.119140	.175	.244140	.275	.369140	.375	.494140
.076	.121093	.176	.246093	.276	.371093	.376	.496093
.077	.123046	.177	.248046	.277	.373046	.377	.498046

OCTAL/DECIMAL FRACTION CONVERSIONS

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000000	.000000	.000100	.000244	.000200	.000488	.000300	.000732
.000001	.000003	.000101	.000247	.000201	.000492	.000301	.000736
.000002	.000007	.000102	.000251	.000202	.000495	.000302	.000740
.000003	.000011	.000103	.000255	.000203	.000499	.000303	.000743
.000004	.000015	.000104	.000259	.000204	.000503	.000304	.000747
.000005	.000019	.000105	.000263	.000205	.000507	.000305	.000751
.000006	.000022	.000106	.000267	.000206	.000511	.000306	.000755
.000007	.000026	.000107	.000270	.000207	.000514	.000307	.000759
.000010	.000030	.000110	.000274	.000210	.000518	.000310	.000762
.000011	.000034	.000111	.000278	.000211	.000522	.000311	.000766
.000012	.000038	.000112	.000282	.000212	.000526	.000312	.000770
.000013	.000041	.000113	.000286	.000213	.000530	.000313	.000774
.000014	.000045	.000114	.000289	.000214	.000534	.000314	.000778
.000015	.000049	.000115	.000293	.000215	.000537	.000315	.000782
.000016	.000053	.000116	.000297	.000216	.000541	.000316	.000785
.000017	.000057	.000117	.000301	.000217	.000545	.000317	.000789
.000020	.000061	.000120	.000305	.000220	.000549	.000320	.000793
.000021	.000064	.000121	.000308	.000221	.000553	.000321	.000797
.000022	.000068	.000122	.000312	.000222	.000556	.000322	.000801
.000023	.000072	.000123	.000316	.000223	.000560	.000323	.000805
.000024	.000076	.000124	.000320	.000224	.000564	.000324	.000808
.000025	.000080	.000125	.000324	.000225	.000568	.000325	.000812
.000026	.000083	.000126	.000328	.000226	.000572	.000326	.000816
.000027	.000087	.000127	.000331	.000227	.000576	.000327	.000820
.000030	.000091	.000130	.000335	.000230	.000579	.000330	.000823
.000031	.000095	.000131	.000339	.000231	.000583	.000331	.000827
.000032	.000099	.000132	.000343	.000232	.000587	.000332	.000831
.000033	.000102	.000133	.000347	.000233	.000591	.000333	.000835
.000034	.000106	.000134	.000350	.000234	.000595	.000334	.000839
.000035	.000110	.000135	.000354	.000235	.000598	.000335	.000843
.000036	.000114	.000136	.000358	.000236	.000602	.000336	.000846
.000037	.000118	.000137	.000362	.000237	.000606	.000337	.000850
.000040	.000122	.000140	.000366	.000240	.000610	.000340	.000854
.000041	.000125	.000141	.000370	.000241	.000614	.000341	.000858
.000042	.000129	.000142	.000373	.000242	.000617	.000342	.000862
.000043	.000133	.000143	.000377	.000243	.000621	.000343	.000865
.000044	.000137	.000144	.000381	.000244	.000625	.000344	.000869
.000045	.000141	.000145	.000385	.000245	.000629	.000345	.000873
.000046	.000144	.000146	.000389	.000246	.000633	.000346	.000877
.000047	.000148	.000147	.000392	.000247	.000637	.000347	.000881
.000050	.000152	.000150	.000396	.000250	.000640	.000350	.000885
.000051	.000156	.000151	.000400	.000251	.000644	.000351	.000888
.000052	.000160	.000152	.000404	.000252	.000648	.000352	.000892
.000053	.000164	.000153	.000408	.000253	.000652	.000353	.000896
.000054	.000167	.000154	.000411	.000254	.000656	.000354	.000900
.000055	.000171	.000155	.000415	.000255	.000659	.000355	.000904
.000056	.000175	.000156	.000419	.000256	.000663	.000356	.000907
.000057	.000179	.000157	.000423	.000257	.000667	.000357	.000911
.000060	.000183	.000160	.000427	.000260	.000671	.000360	.000915
.000061	.000186	.000161	.000431	.000261	.000675	.000361	.000919
.000062	.000190	.000162	.000434	.000262	.000679	.000362	.000923
.000063	.000194	.000163	.000438	.000263	.000682	.000363	.000926
.000064	.000198	.000164	.000442	.000264	.000686	.000364	.000930
.000065	.000202	.000165	.000446	.000265	.000690	.000365	.000934
.000066	.000205	.000166	.000450	.000266	.000694	.000366	.000938
.000067	.000209	.000167	.000453	.000267	.000698	.000367	.000942
.000070	.000213	.000170	.000457	.000270	.000701	.000370	.000946
.000071	.000217	.000171	.000461	.000271	.000705	.000371	.000949
.000072	.000221	.000172	.000465	.000272	.000709	.000372	.000953
.000073	.000225	.000173	.000469	.000273	.000713	.000373	.000957
.000074	.000228	.000174	.000473	.000274	.000717	.000374	.000961
.000075	.000232	.000175	.000476	.000275	.000720	.000375	.000965
.000076	.000236	.000176	.000480	.000276	.000724	.000376	.000968
.000077	.000240	.000177	.000484	.000277	.000728	.000377	.000972

OCTAL/DECIMAL FRACTION CONVERSIONS

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000400	.000976	.000500	.001220	.000600	.001464	.000700	.001708
.000401	.000980	.000501	.001224	.000601	.001468	.000701	.001712
.000402	.000984	.000502	.001228	.000602	.001472	.000702	.001716
.000403	.000988	.000503	.001232	.000603	.001476	.000703	.001720
.000404	.000991	.000504	.001235	.000604	.001480	.000704	.001724
.000405	.000995	.000505	.001239	.000605	.001483	.000705	.001728
.000406	.000999	.000506	.001243	.000606	.001487	.000706	.001731
.000407	.001003	.000507	.001247	.000607	.001491	.000707	.001735
.000410	.001007	.000510	.001251	.000610	.001495	.000710	.001739
.000411	.001010	.000511	.001255	.000611	.001499	.000711	.001743
.000412	.001014	.000512	.001258	.000612	.001502	.000712	.001747
.000413	.001018	.000513	.001262	.000613	.001506	.000713	.001750
.000414	.001022	.000514	.001266	.000614	.001510	.000714	.001754
.000415	.001026	.000515	.001270	.000615	.001514	.000715	.001758
.000416	.001029	.000516	.001274	.000616	.001518	.000716	.001762
.000417	.001033	.000517	.001277	.000617	.001522	.000717	.001766
.000420	.001037	.000520	.001281	.000620	.001525	.000720	.001770
.000421	.001041	.000521	.001285	.000621	.001529	.000721	.001773
.000422	.001045	.000522	.001289	.000622	.001533	.000722	.001777
.000423	.001049	.000523	.001293	.000623	.001537	.000723	.001781
.000424	.001052	.000524	.001296	.000624	.001541	.000724	.001785
.000425	.001056	.000525	.001300	.000625	.001544	.000725	.001789
.000426	.001060	.000526	.001304	.000626	.001548	.000726	.001792
.000427	.001064	.000527	.001308	.000627	.001552	.000727	.001796
.000430	.001068	.000530	.001312	.000630	.001556	.000730	.001800
.000431	.001071	.000531	.001316	.000631	.001560	.000731	.001804
.000432	.001075	.000532	.001319	.000632	.001564	.000732	.001808
.000433	.001079	.000533	.001323	.000633	.001567	.000733	.001811
.000434	.001083	.000534	.001327	.000634	.001571	.000734	.001815
.000435	.001087	.000535	.001331	.000635	.001575	.000735	.001819
.000436	.001091	.000536	.001335	.000636	.001579	.000736	.001823
.000437	.001094	.000537	.001338	.000637	.001583	.000737	.001827
.000440	.001098	.000540	.001342	.000640	.001586	.000740	.001831
.000441	.001102	.000541	.001346	.000641	.001590	.000741	.001834
.000442	.001106	.000542	.001350	.000642	.001594	.000742	.001838
.000443	.001110	.000543	.001354	.000643	.001598	.000743	.001842
.000444	.001113	.000544	.001358	.000644	.001602	.000744	.001846
.000445	.001117	.000545	.001361	.000645	.001605	.000745	.001850
.000446	.001121	.000546	.001365	.000646	.001609	.000746	.001853
.000447	.001125	.000547	.001369	.000647	.001613	.000747	.001857
.000450	.001129	.000550	.001373	.000650	.001617	.000750	.001861
.000451	.001132	.000551	.001377	.000651	.001621	.000751	.001865
.000452	.001136	.000552	.001380	.000652	.001625	.000752	.001869
.000453	.001140	.000553	.001384	.000653	.001628	.000753	.001873
.000454	.001144	.000554	.001388	.000654	.001632	.000754	.001876
.000455	.001148	.000555	.001392	.000655	.001636	.000755	.001880
.000456	.001152	.000556	.001396	.000656	.001640	.000756	.001884
.000457	.001155	.000557	.001399	.000657	.001644	.000757	.001888
.000460	.001159	.000560	.001403	.000660	.001647	.000760	.001892
.000461	.001163	.000561	.001407	.000661	.001651	.000761	.001895
.000462	.001167	.000562	.001411	.000662	.001655	.000762	.001899
.000463	.001171	.000563	.001415	.000663	.001659	.000763	.001903
.000464	.001174	.000564	.001419	.000664	.001663	.000764	.001907
.000465	.001178	.000565	.001422	.000665	.001667	.000765	.001911
.000466	.001182	.000566	.001426	.000666	.001670	.000766	.001914
.000467	.001186	.000567	.001430	.000667	.001674	.000767	.001918
.000470	.001190	.000570	.001434	.000670	.001678	.000770	.001922
.000471	.001194	.000571	.001438	.000671	.001682	.000771	.001926
.000472	.001197	.000572	.001441	.000672	.001686	.000772	.001930
.000473	.001201	.000573	.001445	.000673	.001689	.000773	.001934
.000474	.001205	.000574	.001449	.000674	.001693	.000774	.001937
.000475	.001209	.000575	.001453	.000675	.001697	.000775	.001941
.000476	.001213	.000576	.001457	.000676	.001701	.000776	.001945
.000477	.001216	.000577	.001461	.000677	.001705	.000777	.001949

APPENDIX F - STANDARD CHARACTER CODES

Symbol	ASCII	Printer	Mag Tape	Hollerith	FORTRAN
@	300	00	32	0-2-8	77
A	301	01	61	12-1	13
B	302	02	62	12-2	14
C	303	03	63	12-3	15
D	304	04	64	12-4	16
E	305	05	65	12-5	17
F	306	06	66	12-6	20
G	307	07	67	12-7	21
H	310	10	70	12-8	22
I	311	11	71	12-9	23
J	312	12	41	11-1	24
K	313	13	42	11-2	25
L	314	14	43	11-3	26
M	315	15	44	11-4	27
N	316	16	45	11-5	30
O	317	17	46	11-6	31
P	320	20	47	11-7	32
Q	321	21	50	11-8	33
R	322	22	51	11-9	34
S	323	23	22	0-2	35
T	324	24	23	0-3	36
U	325	25	24	0-4	37
V	326	26	25	0-5	40
W	327	27	26	0-6	41
X	330	30	27	0-7	42
Y	331	31	30	0-8	43
Z	332	32	31	0-9	44
[	333	33	75	12-5-8	76**
\	334	34	36	0-6-8	76**
]	335	35	55	11-5-8	76**
!	336	36	17*	7-8	76**
-	337	37	20	2-8	76***
(blank)	240	40	20	No punch	00
!	241	41	52	11-2-8	51
"	242	42	35	0-5-8	62

STANDARD CHARACTER CODES

Symbol	ASCII	Printer	Mag Tape	Hollerith	FORTRAN
#	243	43	37	0-7-8	63
\$	244	44	53	11-3-8	60
%	245	45	57	11-7-8	64
&	246	46	77	12-7-8	65
'	247	47	14	4-8	66
(	250	50	34	0-4-8	52
)	251	51	74	12-4-8	53
*	252	52	54	11-4-8	47
+	253	53	60	12	45
,	254	54	33	0-3-8	54
-	255	55	40	11	46
.	256	56	73	12-3-8	51
/	257	57	21	0-1	50
0	260	60	12	0	01
1	261	61	01	1	02
2	262	62	02	2	03
3	263	63	03	3	04
4	264	64	04	4	05
5	265	65	05	5	06
6	266	66	06	6	07
7	267	67	07	7	10
8	270	70	10	8	11
9	271	71	11	9	12
:	272	72	15	5-8	67
;	273	73	56	11-6-8	70
<	274	74	76	12-6-8	76**
=	275	75	13	3-8	55
>	276	76	16	6-8	76***
?	277	77	72	12-2-8	76

* End of file for magnetic tape.

** Undefined character (FORTRAN).

*** Form control -- return to column 1 (FORTRAN).

**** Tab control -- skip to column 7 (FORTRAN).

APPENDIX G - TELETYPE CHARACTER CODES

Character	Internal Code	Character	Internal Code
0	260	P	320
1	261	Q	321
2	262	R	322
3	263	S	323
4	264	T	324
5	265	U	325
6	266	V	326
7	267	W	327
8	270	X	330
9	271	Y	331
A	301	Z	332
B	302	(blank)	240
C	303	!	241
D	304	"	242
E	305	#	243
F	306	\$	244
G	307	%	245
H	310	&	246
I	311	'	247
J	312	(	250
K	313	)	251
L	314	*	252
M	315	+	253
N	316	,	254
O	317	-	255

TELETYPE CHARACTER CODES

Character	Internal Code	Character	Internal Code
.	256	LINE FEED	212
/	257	V TAB	213
:	272	FORM	214
;	273	RETURN	215
<	274	SO	216
=	275	SI	217
>	276	DCO	220
?	277	X-ON	221
@	300	TAPE AUX	
[	333	ON	222
\	334	X-OFF	223
]	335	TAPE OFF	
!	336	AUX	224
-	337	ERROR	225
RUBOUT	377	SYNC	226
NUL	200	LEM	227
SOM	201	S0	230
EOA	202	S1	231
EOM	203	S2	232
EOT	204	S3	233
WRU	205	S4	234
RU	206	S5	235
BELL	207	S6	236
FE	210	S7	237
H TAB	211		

NOTES

NOTES

(please detach card here)

First Class
Permit No. 323
Newport Beach
California

BUSINESS REPLY MAIL No Postage Stamp Necessary if Mailed in United States

postage will be paid by



varian data machines
2722 michelson drive
irvine/california/92664



varian data machines

2722 michelson drive
irvine/california/92664

I AM INTERESTED IN:

- ☐ Varian 620/L-100
☐ Varian 620/f-100
☐ Varian ADAPTS with Extended BASIC
☐ Varian 620/L Computer
☐ STATOS Electrostatic Printer/Plotter
☐ Varian R 620 Ruggedized Computer
☐ Analog-to-Digital Converters
☐ Varian 620/DC Data Communication System
☐ VORTEX ☐ BEST ☐ RPG/MOS ☐ PERT

APPLICATION:

- ____ Oceanography
____ Batch processing
____ Real Time Operating System
____ Analytical instrumentation data
____ Telemetry or aerospace test data
____ Small computer used as a local or as a remote satellite to a larger computer for control of I/O peripherals
____ Numerical control of machine tools or drafting machines
____ Control of chemical, petroleum, or power generation processes
____ Communication switching, data concentration, etc.
____ Production line testing
____ Medical
____ Scientific (Lab/R & D)
____ Other _____

please specify

- ☐ Please have a sales engineer call me
Area Code _____ Tel. _____ Ext. _____
☐ Please add my name to your mailing list

Mail to:

NAME _____

TITLE _____

COMPANY _____ DEPT. CODE _____

ADDRESS _____

CITY _____

STATE _____ ZIP _____

\$3.00

varian
620 100's
computer
handbook



varian data machines
2722 Michelson Drive
Irvine, California 92664